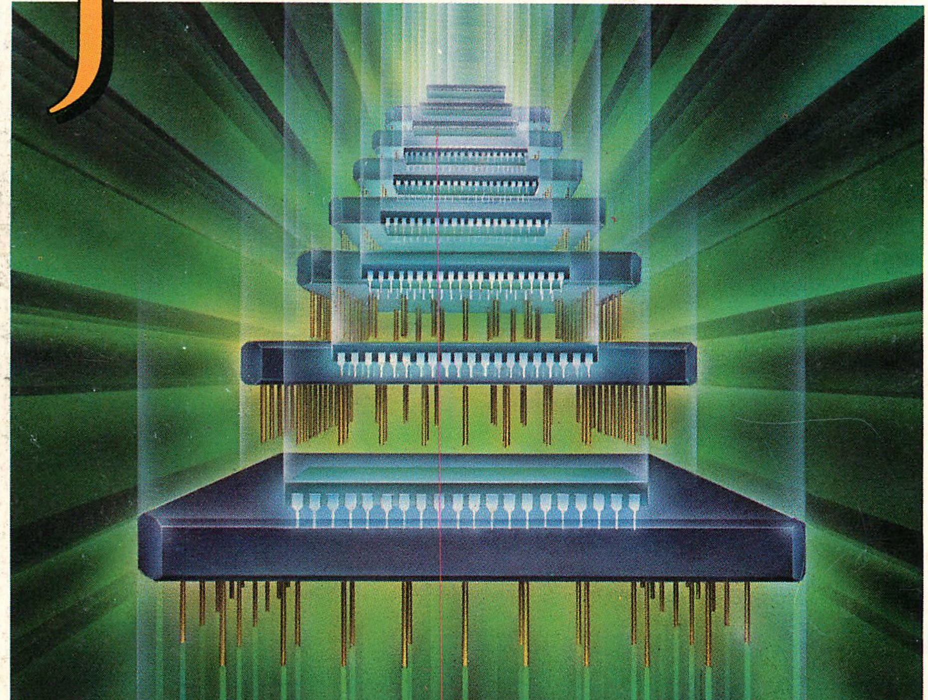
OCTOBER 1985 VOL. 3, No. 10 $3.95

FOR IBM PERSONAL COMPUTER USERS

PC TECH JOURNAL



MACRO ASSEMBLERS

IBM and Microsoft Support 286 Family

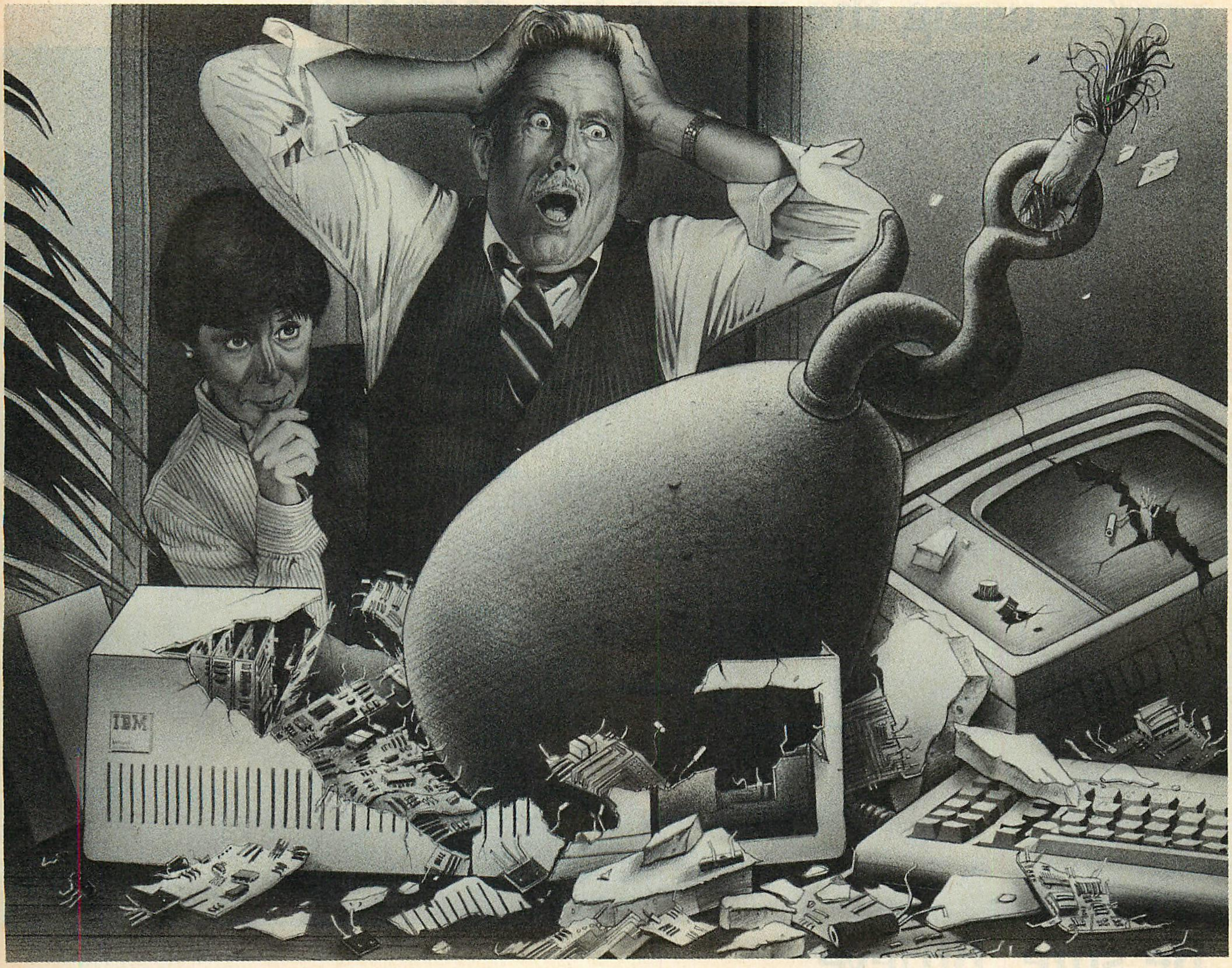
OBJECT MODULES EXPLAINED

DATA MANAGER REPORT: ZIM

KEYBOARD MACRO TOOLS

0 14024 14357 10

My accounts receivable are under what???



What would it mean to you if the valuable data files stored on your computer disk were suddenly erased or destroyed?

A wrecking ball? Not likely! But every day vital records are destroyed by operator error, power failures, vandalism, fire or some other accident.

Why take a chance when it is so inexpensive and easy to protect your records with an Irwin BACKUP™ Tape System. Simply plug it into your IBM PC/XT, AT or compatible microcomputer and the Irwin BACKUP™ Tape System is ready to go to work.

Easy to use. The power of one finger activates the Irwin BACKUP™ Tape System. In a few unattended minutes, on a data cartridge so small it fits easily in a shirt pocket, the Irwin BACKUP™ Tape System automatically preserves and verifies the information stored on a hard disk. Your vital records are safe, out of harm's way.

With almost one hundred thousand systems shipped, Irwin, the leader in Tape Backup Systems for microcomputers, knows your needs in backup protection and convenience.

INTEGRITY - It's critical that a backup system be accurate.

EC/TAPE™ is a unique automatic error correction system which assures you of data integrity.

INTERCHANGEABILITY - A backup tape must be capable of being reliably read from computer to computer.

Irwin's patented track following feature assures that your backup tape will be readable on any computer equipped with an Irwin BACKUP™ Tape System.

COMPLETE/FRIENDLY SOFTWARE - Backup procedures must be simple and convenient to use or they won't be used consistently.

Irwin's software permits unattended backup of your entire disk, or only those files you select. It's complete, easy to understand, and easy to use.

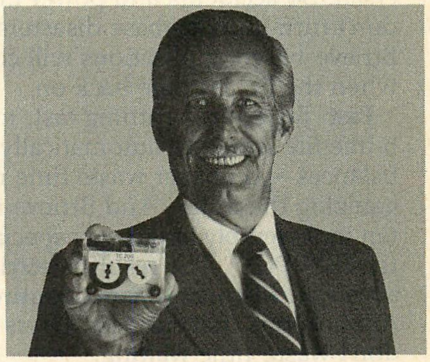
Don't wait for a problem. Protection is easy and inexpensive with Irwin. For more information, call **1-800/421-1879**, or stop in for a "SHOW ME" demonstration at a leading computer store near you.

CIRCLE NO. 164 ON READER SERVICE CARD

BACKUP

FAST, EASY, INEXPENSIVE.

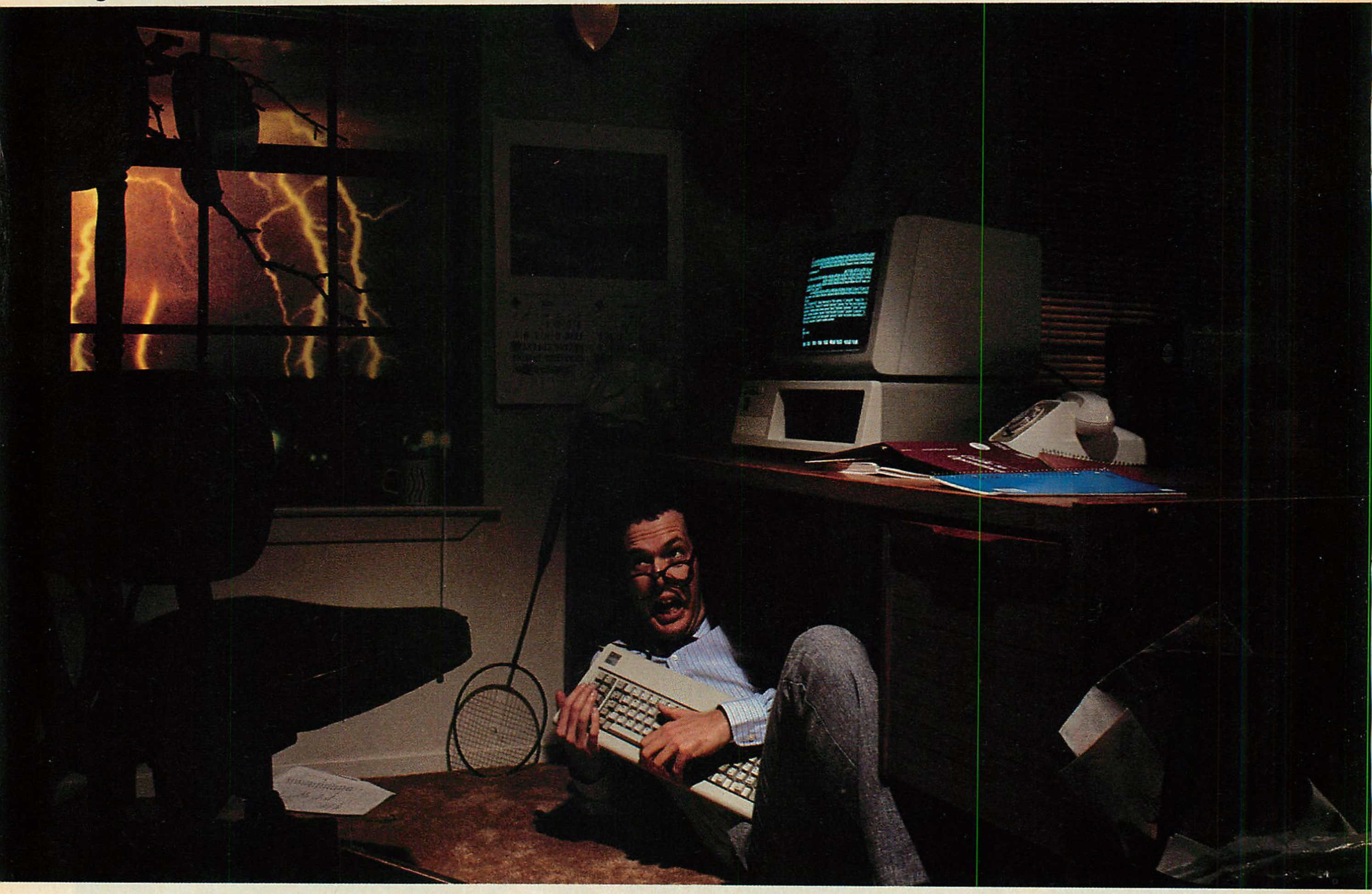
"No Problem, Thanks Irwin."



IRWIN™ MAGNETICS

2311 Green Road
Ann Arbor, Michigan 48105
313/996-3300 1-800/421-1879

If lightning still scares you, you're using the wrong file manager.



Be sure. Btrieve™.

Lightning may strike. But it doesn't have to destroy your database.

Btrieve™ file management offers automatic file recovery after a system crash. So accidents and power failures don't turn into database disasters. Your Btrieve-based applications will come up when the lights come back on.

Fast. Btrieve is lightning fast, too. Its b-tree file structure automatically balances—you never waste time reorganizing the index. And Btrieve is written in Assembly language especially for the IBM PC™. The result: electrifying speed on file maintenance routines. Applications that run fast. Users who don't waste time waiting.

The standard for networking. When your application requires multi-user file sharing, Btrieve/N (network version) sets the standard for the industry's most popular LANs: IBM's PC Network, Netware™, Davong MultiLink™, Omninet™, PC Net™, EtherSeries™, Nestar™, and NetOne™. Btrieve/N offers **safe** network file management that coordinates simultaneous updates and protects against lost data.

Fully-relational data management. Using SoftCraft's entire family of products gives you a complete, fully-relational database management system. Rtrieve™ adds report writing capabilities for generating the reports you need. Xtrieve™ speeds users through database queries with interactive, on-screen menus—no command language or special syntax.

For professional programmers. Btrieve is the fast, reliable answer for all your application development. In any development language—BASIC, Pascal, Cobol, C, Fortran, and APL. With multikey access to records. Unlimited records per file. Duplicate, modifiable, and segmented keys. Variable cache buffer.

With Btrieve, you can develop better applications faster. And know they'll be safe if lightning strikes.

Suggested retail prices: Btrieve, $245; Btrieve/N, $595; Xtrieve, $195; Xtrieve/N, $395; Rtrieve, $85; Rtrieve/N, $175. Requires PC-DOS or MS™-DOS 1.X, 2.X, or 3.X.

Btrieve, Xtrieve, and Rtrieve; IBM; Netware; Davong MultiLink; Omninet; PC Net; EtherSeries; Nestar; NetOne; and MS are trademarks of SoftCraft Inc.; International Business Machines; Novell Data Systems; Davong Systems Inc.; Corvus Systems; Orchid Technology; 3Com Corp.; Nestar Systems Inc.; Ungermann-Bass; and Microsoft Inc.

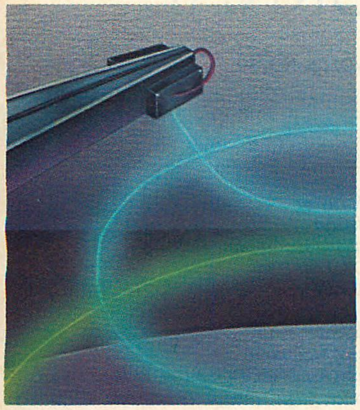
SoftCraft Inc.
P.O. Box 9802 #917
Austin, Texas 78766
(512) 346-8380 Telex 358 200

CIRCLE NO. 201 ON READER SERVICE CARD

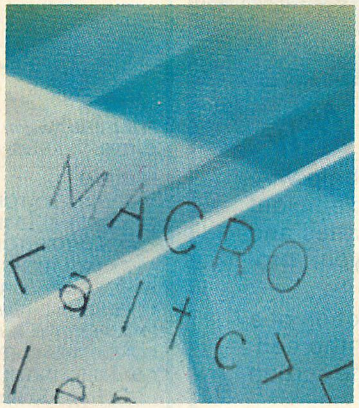
OCTOBER 1985

PC TECH JOURNAL

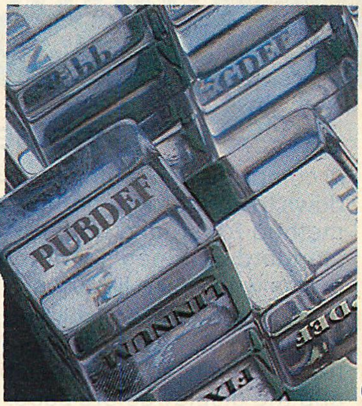
VOLUME 3, NUMBER 10



85



131



62

Cover illustration · Alan Hashimoto

PC TECH JOURNAL

VOL. 3, NO. 10

PUBLISHER: *Newton Barrett*
EDITOR: *Will Fastie*

EDITORIAL
MANAGING EDITOR: *Marjory Spraycar*
EXECUTIVE EDITOR: *Julie Anderson*
TECHNICAL EDITORS: *Jeff Duntemann, Caroline Halliday*
ASSOCIATE TECHNICAL EDITOR: *Dan Beale*
SENIOR COPY EDITOR: *Susan Holly*
COPY EDITOR: *Gail Shaffer*
PROOFREADER: *Kathleen Peddicord*
CONSULTING EDITOR: *Thomas V. Hoffmann*
CONTRIBUTING EDITORS: *Steven Armbrust, Don Awalt, Richard Foard, Ted Forgeron, Arthur A. Gleckler, Augie Hansen, Ted Mirecki, William H. Murray, Max Stul Oppenheimer*
ADMINISTRATIVE ASSISTANT: *Paula Lamberti*
EDITORIAL ASSISTANT: *Carole Autenzio*

ART & PRODUCTION
ART DIRECTOR: *Ina Saltz*
ASSOCIATE ART DIRECTOR: *Sharon Reuter*
ASSISTANT ART DIRECTOR: *Sandra Ray*
ADVERTISING PRODUCTION MANAGER: *Lisa Franey Ducey*
EDITORIAL PRODUCTION COORDINATOR: *David K. Chipps*

ADVERTISING SALES
MARKETING COORDINATOR: *Julie Henderson*
ADVERTISING COORDINATOR: *Michele Fischetti*
DISTRICT MANAGERS: *Rita Burke, Jan Schultz—East Coast; Lisa Kampfmann—Midwest; Ted Bahr, Bill Bush, Phyllis Egan—West Coast*
ACCOUNT REPRESENTATIVES: *Pat Toohey—East Coast; Rosemarie Caruso—Midwest/East Coast; Pam Sigal, Arlene Steadman—West Coast*

CIRCULATION
CIRCULATION MANAGER: *Shane Boel*
CIRCULATION SALES DEVELOPMENT: *Daniel Rosensweig*
MEDIA MANAGER: *Melinda Kendall*

COMPUTER PUBLICATIONS DIVISION
PRESIDENT: *Kenneth H. Koppel*
VICE PRESIDENT, Editorial: *Jonathan D. Lazarus*
VICE PRESIDENT, Production: *Baird Davis*
VICE PRESIDENT, Creative Services: *Herbert Stern*
VICE PRESIDENT, Circulation: *Alicia Marie Ivans*
VICE PRESIDENT, Circulation Services: *James Ramaley*
VICE PRESIDENT: *Hugh Tietjen*
MARKETING MANAGER: *Ronnie Sonnenberg*
VICE PRESIDENT, Marketing Services: *Ann Pollak Adelman*
BUSINESS MANAGER: *Gary A. Gustafson*
EDITORIAL DIRECTOR: *Ernest F. Baxter*

ZIFF-DAVIS PUBLISHING
PRESIDENT, *Richard P. Friese;* SENIOR VICE PRESIDENT, *Kenneth H. Koppel;* SENIOR VICE PRESIDENT, *Philip Sine;* VICE PRESIDENT, *William L. Phillips;* TREASURER, *Selwyn Taubman;* SECRETARY, *Bertram A. Abrams*

EDITORIAL OFFICE
PC TECH JOURNAL, The World Trade Center, Suite 211, Baltimore, MD 21202. 301/576-0770. MCIMail: PCTECH. PCTECHline: 301/576-PCTJ.

ADVERTISING OFFICES
(East Coast/Midwest) One Park Ave., New York, NY 10016. 212/503-5185. (West Coast) 3460 Wilshire Blvd., Los Angeles, CA 90010. 213/387-2100; 11 Davis Drive, Belmont, CA 94002. 415/598-2290.

SUBSCRIPTION INQUIRIES
PC TECH JOURNAL, P.O. Box 2968, Boulder, CO 80321. Subscription service: 303/447-9330. Back issues: send $7.00/copy to *PC TECH JOURNAL*, Box CN, 1914, Morristown, NJ 07960.

PC TECH JOURNAL (ISSN 0738-0194) is published monthly, $29.97 for one year, $52.97 for two years, $69.97 for three years. Additional postage $12 for Canada & Foreign by Ziff-Davis Publishing Company, One Park Avenue, New York, NY 10016. Second-Class Postage paid at New York, NY and at additional mailing offices. POSTMASTER: Send address changes or subscription inquiries to P.O. Box 2968, Boulder, CO 80321.

BPA
SELECTED MARKET AUDIT DIVISION

Introducing Periscope II

30-Day Money-Back Guarantee

Professional Debugger with Break-out Switch

What Users Say:

"Very powerful for debugging and testing. Great product! Better than Atron by far."
—Wynn Bailey

[Periscope is] "the most essential element of my 'developer's tool box' . . . Every PC used for development at Microstuf has a Periscope board installed and in use on a daily basis . . . Anyone trying to write software on a PC who's not using Periscope is wasting either time, money, or both." —Jeff Garbers

Now there are TWO Periscopes! The original "board" model includes the write-protected RAM board. It is called Periscope I. The new "software" model does not include the write-protected RAM board. It is called Periscope II. We specify 'I' or 'II' only for features not in both models.

Periscope is "Always there with just a push of the button." Install the switch and software, then go about business as usual. Until your system hangs or the keyboard locks up or you just get curious about what's going on . . . then press the break-out switch and Presto! Periscope's debugging power is at your command. When you return to the executing program, it won't even know that Periscope has been there.

Save time with symbols and source code. They give you a roadmap through memory! Periscope uses names—symbols—from your program so you don't have to remember addresses. It displays source code and line numbers from high-level languages, too. You save hours of time because you access what you need with familiar names!

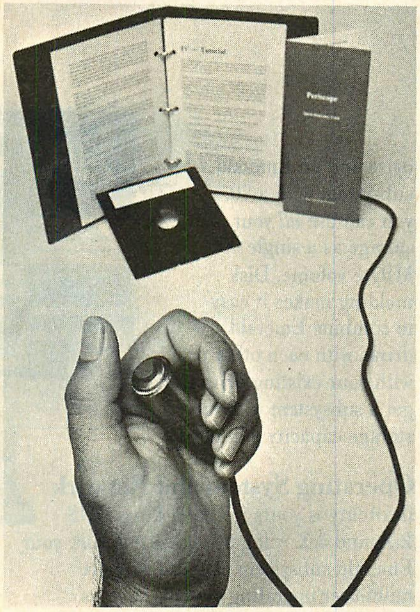
Can debugging be fun? With Periscope it can! Here's why:

It's Fast! Written entirely in assembler to save you time.

It's Easy! Commands similar to Debug's, optional on-line help, quick-reference card, tutorial, 150+ page manual, and support direct from the author—all you need to get up to speed quickly.

It's Flexible! You can:

- Enter Periscope via break-out switch, program loader, or assigned keys
- Define up to four windows for data, register, stack, and disassembly information; change them any time
- Display memory in ASCII, byte, word, double word, integer, or signed integer
- Design your own templates to display memory in easy-to-read formats
- Customize Periscope with your own programs via the user exit capability
- Assign frequently-used commands to function keys



The break-out switch really sets Periscope II apart from typical software-only debuggers. It installs easily, without taking an extra slot.

It's all there! Periscope includes all the standard debugger features, plus extras like supporting the use of one or two monitors, enabling you to search for address references, etc. The latest version includes new and enhanced features to help you debug your programs faster and easier than ever before:

New!	In-line Symbolic Assembler
Enhanced!	75+ Breakpoint Options
New!	Traceback
Enhanced!	Optional Windows
New!	Source-Level Debug
New!	View text files
New!	EGA Support
New!	User Exits
New!	8087/80287 Status
	Much More!

Periscope's unique breakpoint power forces bugs out from where they hide! With over 75 breakpoint options, including both temporary and sticky code breakpoints, you'll find elusive bugs fast. For instance, you can break on register, byte and word values; stop on execution of source lines and interrupts; break on reads and writes to ranges of I/O ports and ranges of memory using various tests. The memory breakpoint is great for debugging C programs with broken pointers. You can even write your own custom breakpoint tests!

Periscope requires: An IBM PC, XT, AT or close compatible; DOS 2.0 or later; 128K RAM; Disk Drive; an 80-column Monitor.

What Reviewers Say:

"This product is a reliable and useful tool for any programmer's workshop. A number of innovative, helpful features are yours for the taking." —Programmer's Journal

"Periscope has excellent on-line help. We were impressed by Periscope's very fast response in all its operations. It is a pleasure to use, and a refreshingly different product . . . offers great value and unique advantages." —Boston Computer Society

Debug any program, any time! Periscope is, in one user's words, "Robust". Use it to debug almost any program, even device drivers, memory-resident, and non-DOS programs. Debug when DOS is broken; debug DOS. Periscope won't let you down when you need it.

User calls Periscope I "Bulletproof"! The installation program in Periscope I loads the crucial debugger software into the RAM board and write-protects it. No runaway program can touch this code! Coupled with the break-out capability and the built-in reliability of the software, this protection gives you the most powerful crash recovery system available.

"Outstanding value for the services offered" writes one user. ***"Worth the price!"*** writes another. With the 30-Day Money-Back Guarantee, you can't lose! If you're not satisfied with Periscope for any reason, just return it within 30 days of receipt for a full refund.

Periscope I: Board, Switch, Software, Manual, Reference Card . . Just $295.00

Periscope II: Switch, Software, Manual, Reference Card . . . . Only $145.00

The U.S. Navy gets Periscopes from us . . . shouldn't you? Order today!

For Immediate Shipment Or More Information, Call Toll-Free

800-722-7006

Get Your Programs Up and Running;

UP PERISCOPE!

DATA BASE DECISIONS
14 Bonnie Lane • Atlanta, GA 30328
404/256-3860

CIRCLE NO. 215 ON READER SERVICE CARD

AT Powe

High performance, high capacity hard disk subsystems

A Single DOS Volume up to 240 formatted MBytes in size on your PC is only the beginning of Emerald subsystems' features. However, it is a very important one to many of our customers.

Most serious micro computer users are familiar with the infamous *32 MByte DOS Barrier*. Some of you have only read about it; others have run up against it head on when trying to run PC Focus or a downloaded mainframe program.

If you are one of those who have hit it head on you'll be pleased to know that *for almost a year Emerald has been shipping fixed disk subsystems that solve your problem*. Subsystems. Plural.

All Emerald subsystems can be operated as a single volume up to their formatted capacity. That includes our 36, 50, 70 and 140 MByte subsystems, in addition to the 280.

```
C>chkdsk

237633536 bytes total disk space
    65536 bytes in 2 hidden files
    49152 bytes in 2 user files
237518848 bytes available on disk

   262144 bytes total memory
   205152 bytes free
```

Actual printout of CHKDSK on 240 MByte volume.

You Determine how many volumes exist in your PC, and what size they are. You can have as many as *24 volumes*, and make each one exactly the size it needs to be.

Set-up is menu driven and as simple as *"How many do you want"* and *"How big should this one be?"*

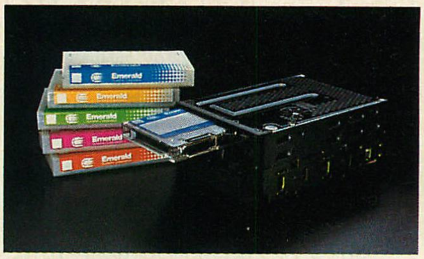
Integrate your existing hard drive into your new sybsystem. Emerald's Disk-Meld technology makes it possible for your XT's 10 Mbyte, or AT's 20 Mbyte drive to become part of a single large volume. For example, if you have an AT with a 20 MByte drive and an Emerald subsystem with a 70, you can use *all* your storage as a single 90 MByte volume. Disk melding makes it easy to combine Emerald drives with each other or with your existing drive to get a subsystem with the storage capacity you need.

Operating System and Network flexibility is yours for the asking. DOS 2.X, and 3.X will get you started. But, your Emerald subsystem can also support multi-user operating systems such as XENIX, VENIX, QNX and PC/IX and networks such as Novell, Sytek, Ungermann-Bass, 3 COM, X-Net, 10 Net, DNA Systems…

Additional Features include password security, 30 millisecond average access time, *automatic retreat to a safe landing zone* in the event of power failure or shutdown, and a long list of PC and AT compatible micro computers.

High performance $^1/_4$" tape backup

60 MBytes in 12 minutes is *FAST* backup, but there's more. If your files are larger than 60 MBytes, Emerald's Backup and Restore Utility (BRU) software will automatically break your file into 60 MByte sections and prompt you for a new cartridge. Of course, restoring is just as easy.



Compact tape drive fits in the AT's front panel expansion space. 60 MByte cartridges are certified for high performance and supplied with color coded labels.

Emerald subsystems were designed for the PC, AT and compatibles such as the AT&T 6300 and Compaq DeskPro.

Menu driven software makes it simple, even for novices, to backup or restore exactly what is needed, and no more. Choose one or more files that were modified after a *Specified Date and Time*, one or more *Specific Files or Directories*, or *All Files and Directories on a DOS Logical Volume*.

Restore data on a different micro if you like. Backup up your company's Emerald subsystem in Portland, Maine and ship the tape to your office in Los Angeles. Because of the defect mapping technique used on Emerald hard disks, the subsystem in L.A. will import your data error free.

Special defect mapping technique allows data to be restored on subsystems other than the original source.

The BRU software *automatically checks, and adjusts to, the defect map* of every Emerald subsystem before restoring data to it.

IBM PC, XT, AT, & PC/IX™ International Business Machines Corp.—AT&T™ American Telephone and Telegraph Corp.—Compaq & DeskPro™ Compaq Computer Corp.—Ungermann-Bass™ Ungermann-Bass, Inc. — Emerald, DiskMeld, BRU & Mainframe Storage for Micros™ Emerald Systems Corp.—UPS™ United Parcel Service.—3 COM™ 3 COM Corp.—DNA Systems™ Network Development Corp.—PC Focus™ Information

r is Yours.

Configured for today's Micros

The Physical Design of the Emerald subsystems lets you determine the configuration that will best serve your needs. Many of today's high performance micros have sacrificed expansion space in favor of compact size; others offer plenty of room for additional drives, tape units and expansion cards. Emerald subsystems provide you with the expansion ability you need. And, they're designed to allow you to continue to expand as your needs continue to grow. No matter which PC you have, there is an Emerald subsystem that will meet your needs.

Subsystems are available for the PC, AT and true compatibles in a variety of configurations.

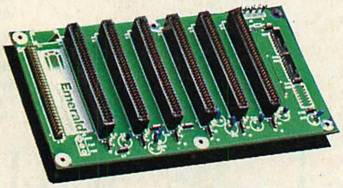
Internal Expansion is easy on the IBM AT and XT. Emerald subsystems are pre-initialized and pre-formatted—just slide the tape or hard drive you have selected into one of the existing expansion areas, plug in a couple of cables, tighten a few screws and replace the system cover. Elapsed time: 10/15 minutes.

The AT accepts 280 MBytes *or* any single drive up to 140 MBytes *and* a 1/4″ tape drive. The XT accepts any Emerald hard drive up to 140 MBytes in size, *or* the 1/4″ tape drive.

External and Portable expansion is easy and practical with the Emerald Portable Subsystems. These IBM color-matched subsystems have their own power supply and are available with hard drives up to 140 MBytes in capacity *or* with a 1/4″ tape drive.

Ordering a tape host adapter card for each of your PCs and physically moving the 1/4″ portable tape sybsystem to where it is needed will save you *thousands of dollars* over the cost of individual portables for each micro.

The portable hard drive configuration is ideal for security sensitive environments. The hard disk is password protected and the entire subsystem is small enough to be locked in a standard safe or filing cabinet.



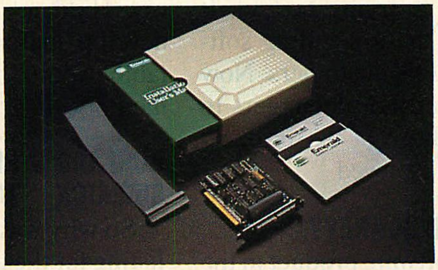
"Expansion chassis" subsystem also provides 6 additional spaces to add cards to your PC.

The 6 Expansion Slots in the Emerald expansion subsystems will be a welcome addition to many micros. If you don't have enough expansion room in your micro, or, if you've used every available slot, then one of these subsystems is just right for you. They are closely matched to the IBM PC in size and color, and have their own, built-in, power supply.

Expansion subsystems are available with drive sizes to 280 MBytes in capacity, with, or without, a built-in 1/4″ tape drive. When coupled with an AT a truly powerful computer system results.

The *real* backup procedure

Standing Behind You all the way is the company that broke the 32 MByte DOS barrier.



Documentation, host adapter card, software and cables are included with each subsystem.

First, we provide everything needed to get you up-and-running quickly. Then, we back you up with specialists, available 16 hours a day, to help with network implementation, applications support, XENIX and technical questions, and to provide *48 hour in-and-out repair service.*

Delivery and Warranty are important considerations. Your Emerald subsystem will be shipped within 14 days of order, via UPS, and comes with a 30 day money back guarantee and a 120 day warranty. Warranty extensions are available for 1 and 2 year periods.

Fast service and fast shipping are company policies.

Users with large database applications in factory automation, CAD/CAM, accounting, medical, R&D, Point-of-Sale and many other areas are already telling their friends about Emerald. A large percentage of them are in Fortune 500 companies, and many others are on their way to Fortune status.

If your company fits one of those categories, or, if you're a smaller company with an application you thought could only be done on a mainframe or mini, pick up the phone. Call Emerald. An Applications Engineer is standing by to answer your questions, send you literature and refer you to an installed site in your area.

Let one of our customers tell you, first hand, how Emerald helped give *his* AT Power.

Emerald
Systems Corporation

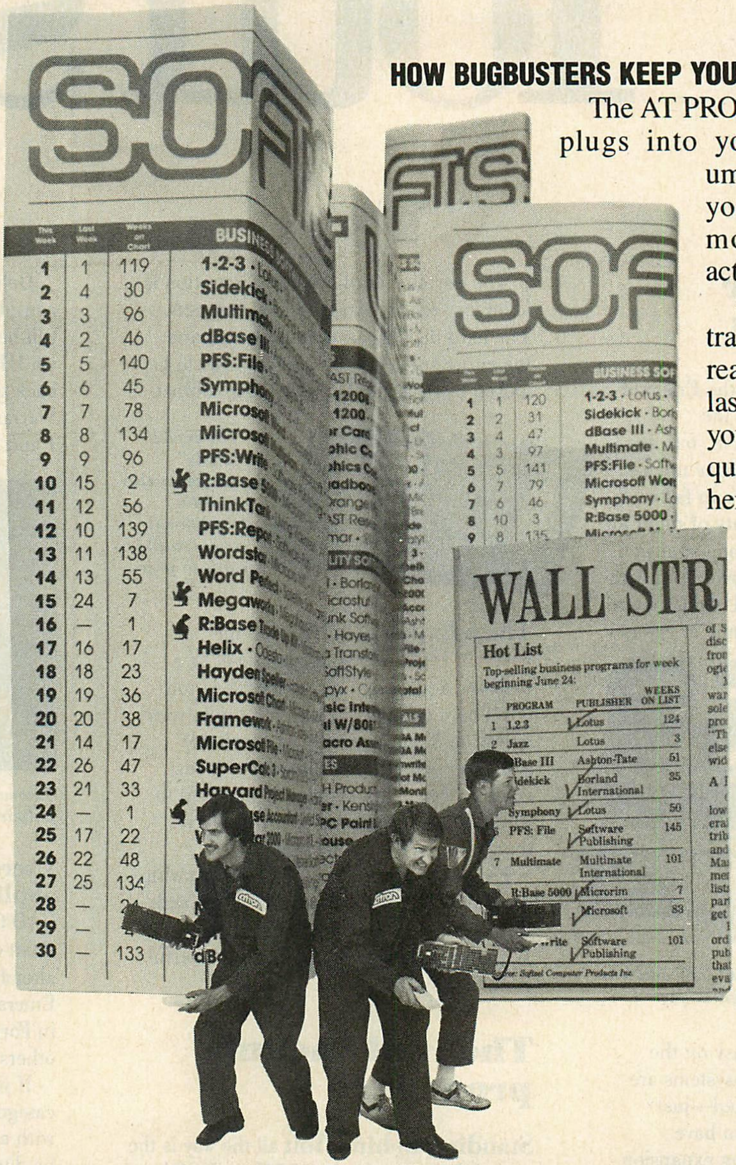
Mainframe Storage for Micros

4757 Morena Blvd.
San Diego, CA 92117
(619) 270-1994
U.S.A.
Telex: 353458 EMERSYS
Easylink: 62853804

CIRCLE NO. 154 ON READER SERVICE CARD

XENIX™ Microsoft Corp.—VENIX™ Ventur Com—QNX™ Quantum Software—Novell™ Novell, Inc.—X-Net™ XCOMP, Inc.—10 Net™ Fox Research, Inc.
Builders.

ATRON'S PC/AT BUGBUSTERS



A BUGBUSTER STORY

Brad Crain, a project manager at Software Publishing (the people who developed both PFS:WRITE and PFS:FILE), relates the following: "On Friday, March 22, 1985, I was about to get on an airplane with Jeff Tucker, who was co-author of PFS:WRITE with me, and fly to IBM's Boca Raton, Florida facility. For a week, we had been unsuccessfully trying to isolate a bug in a new software product. In a last, desperation move, I set up an early-Saturday morning appointment with ATRON.

"Three of us walked through ATRON's door at 8:00 the next morning. Using ATRON's hardware-assisted debugging tools, we had the problem identified and fixed by 10:30AM."

Mr. Crain concludes: "We'd never have found the bug with mere software debuggers, which have the bad habit of getting over-written by the very bugs they're trying to find. It doesn't surprise me that almost all the top-selling software packages were written by ATRON customers. Now that they've broadened their PC family of debuggers to include a PC/AT debugging tool, those of us seriously into 80286 development are greatly relieved."

ARE YOU TRYING TO DO SOMETHING SCAREY?

Like developing your AT-based software product in the dark? Without professional debugging tools?

Seven of the ten top-selling software packages listed by the *THE WALL STREET JOURNAL** were produced by ATRON customers. The PC PROBE™ bugbuster ($1595) accounts for much of this success. Now that the PC/AT is the new standard for advanced commercial and scientific development, ATRON is proud to announce the AT PROBE™ bugbuster ($2495). It has even more debugging capabilities than the PC Probe.

HOW BUGBUSTERS KEEP YOU FROM GETTING SLIMED

The AT PROBE is a circuit board that plugs into your PC/AT. It has an umbilical which plugs into your 80287 socket and monitors all processor activity.

Since AT PROBE can trace program execution in real time, and display the last 2048 memory cycles, you can easily answer the questions: "How did I get here?" and "What *are* the interrupts doing?"

It can solve spooky debugging problems. Like finding where your program over-writes memory or I/O - impossible with software debuggers.

You can even do source-level debugging in your favorite language, like C, Pascal or assembler. And after your application is debugged, the AT PROBE's performance-measurement software can isolate your application's bottlenecks.

Finally, the AT PROBE has its own 1-MByte of memory. Hidden and write-protected. How else could you develop that really large program, where the symbol table would otherwise demand most of your PC/AT memory.

BORLAND'S PHILIPPE KAHN: "THERE WOULDN'T BE A SIDEKICK™ WITHOUT ATRON'S DEBUGGERS."

So why waste more time reading though your program listing for the ten thousandth time, trying to find why your program starts howling with every full moon. Be like BORLAND, get your Atron bugbuster today and bust bugs tomorrow.

Atron
THE DEBUGGER COMPANY
20665 Fourth Street • Saratoga, CA 95070 408/741-5900

Data Interchange

The industry desperately needs standards.

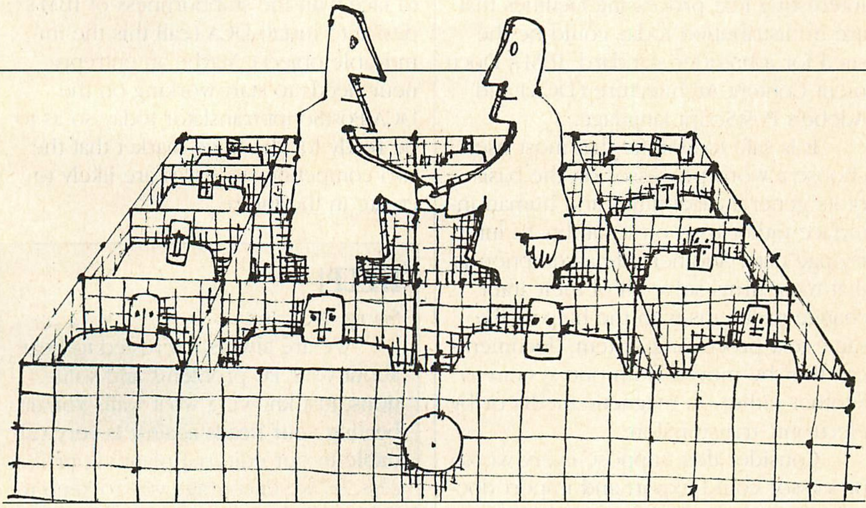


ILLUSTRATION • MACIEK ALBRECHT

There is an axiom in the computer business: adherence to standards is costly. I would like to propose another: ignoring standards is costly.

Technology turns so quickly in this industry that keeping up is often difficult. Failure to get a product to market promptly can spell complete disaster for a company. If compliance to a standard prevents a product from moving quickly out the door, the company may decide to avoid that obstacle.

Compliance to some standards, such as RS-232 or parallel ports, is simple and obvious and usually practiced without question. Other standards, such as those for just about every programming language, are considered open to individual interpretation because the vendor wishes to exploit technology that is more contemporary than the standard. Worst of all, standards remain nonexistent for many critical aspects of computing, either because no one can agree on reasonable specifications or because no one is trying.

Data interchange is the biggest challenge facing software developers today; however, very little work is being done to address the problem in a general way. Development of standards simply is not occurring; no one seems to be researching the nature of data interchange; and vendors have not been able to offer any promising solutions. The overall picture remains an extremely discouraging one.

WHAT'S THE PROBLEM?

What we do hear about these days is *integration*. Vendors of integrated programs tout them as software that performs the major functions desired by most computer users: word processing, spreadsheet, database, and communications. The vendors stress that the programs are all-in-one products, that each function is but part of the whole. As a side issue, they point out that data from one section of the program can be quickly used by another, that the data can, in effect, be interchanged among the various sections of the program.

I do not usually operate any of the integrated packages because none delivers as much capability as the standalone products I use for each of the functional areas. Furthermore, the interchange of data among my programs is not as simple as the cut-and-paste of Framework or Windows, nor is it overly complex; the process often can be sufficiently automated with the help of keyboard macros or other tools.

So what does the integrated package have to offer? The difference between my strategy and that of the integrated program is that I do the data interchange manually (admittedly, it is sometimes a chore) and the integrated program does the work automatically. In fact, the integrated program has an invisible, miniature, standard method of data interchange built in, a method carefully tailored to its needs.

If the functionality of the integrated package is to the user's liking, some of the interchange problems will be addressed. But these tailored solutions leave much to be desired. The capabilities for many operations the user might wish to perform are not provided. In short, the user must stay within the boundaries of the built-in strategy or resort to manual techniques—a possibility the integrated program is supposed to eliminate. Remaining within these boundaries is quite difficult; the need to handle ad hoc situations guarantees that the user eventually will want to go beyond the provided facilities.

One of the reasons integrated programs have emerged is the relative simplicity of building a specialized, internal interchange facility for a captive set of programs. This is not to imply that existing facilities are simple-minded; rather that the complexity of the general interchange problem is enormous by comparison. The developer of an integrated package has the advantage of having all sections of the program under his/her control and thus is able to tailor each section, avoid tricky kinds of interchanges, and make the provided capabilities efficient and effective.

The problem, therefore, is that little science and no standards exist to guide the developer. What is needed is industry-wide agreement about all types of data interchange, beginning with simpler cases and gradually moving to the more complex situations.

TEXT PROCESSING

Text is a good place to start. The computer industry's most in-depth experience is in text processing. Computers

have been doing word processing for two decades or more, and almost every computer ever built in this country sports at least one text processing tool. More important, text processing has long been a favorite computer science study, so a tremendous body of knowledge and information exists on the subject. Xerox (PARC) and Wang most surely have researched text/document interchange and probably have internal standards. Better yet, at least two very interesting text processing facilities that are in distribution today could be the seed for a broader standard: IBM's Document Content Architecture (DCA) and Adobe's PostScript language.

It is safe to assume that most users choose a word processor on the basis of its general capabilities and human interface rather than on its ability to interchange data. Another safe assumption is that most firms solve all of their interchange problems with the help of the same text processing system; documents can thus be moved from one system to another either on magnetic media or by electronic transmission.

Consider this: suppose every word processor could export and import documents in a standard format, perhaps one (like DCA or PostScript) that supported typesetting. Then, given the physical means to move the data, documents could be freely interchanged between vastly different systems, printed on vastly different printers, or typeset by any vendor's equipment.

The power of DCA or PostScript demonstrates that the technology for this class of data interchange exists today. What does not exist is the industry's agreement that adherence to one of these standards, or some other evolutionary one, is desirable. Few word processors currently include either DCA or PostScript, although some have DCA and at least one soon will have both. Very few typesetting services accept ASCII, much less DCA or PostScript (a particularly irritating fact to a magazine). I must emphasize again that industry-wide work on this problem seems to be nonexistent.

CLASH OF STANDARDS

Further complicating the emergence of a standard could be an eventual clash between the two existing text standards. The architecture for IBM's DCA has been around for awhile, is well-documented, and is clearly IBM's spearhead. IBM uses the technology internally and has quietly included it in a number of products. Therefore, the company is likely to do everything it can to perpetuate DCA as a standard. Unfortunately, however, IBM has not been active in bringing its standard into the public light for discussion and consideration.

In contrast, PostScript is very visible in Apple's LaserWriter, and Adobe is buzzing with the activity of other OEM work. The emergence of a de facto standard as a result of a groundswell of support for PostScript (call this the irresistible force) eventually will come face to face with the stubbornness of IBM's desire to install DCA (call this the immovable object). Maybe an entrepreneur needs to start working on the DCA/PostScript translator today so as to be ready for the huge market that the two competing standards are likely to create in the future.

Text is simple when compared to the complexity of interchanges of the structured data found in spreadsheet and database files. In turn, those are simple when compared to the task of interchanging the semantic content (see "The Problem of Computer Semantics," Directions, November/December 1983, p. 10) of any kind of data, which leads to dealing with natural language and its inherent ambiguities. An enormous amount of work still is required.

The microcomputer industry must begin to address these issues through cooperation and research. The industry certainly will benefit from the presence of standards that permit vendors to concentrate on the product at hand rather than agonize over a complex set of low-level details.

HELP!

Sorry. We can't.

We are always interested to hear about your PC problems and solutions. Just knowing what walls you are beating your heads against is very valuable in our editorial planning process, so we encourage you to keep us informed about your problems and discoveries. We read all your letters, and many inspire research and subsequent editorial coverage.

What we cannot do, however, is act as a consulting service. We simply do not have the time to reply to the dozens of written requests for assistance we receive each month, much less answer the telephone calls for our help. We want to be courteous, but we cannot even take the time to reply by mail, stamped, self-addressed envelopes notwithstanding.

We are busy putting out a complex editorial product. We are not a consulting service. Please understand, and help us concentrate on our main goal—publishing *PC Tech Journal*.

PRODUCT OF THE MONTH: IT'S FOR REAL

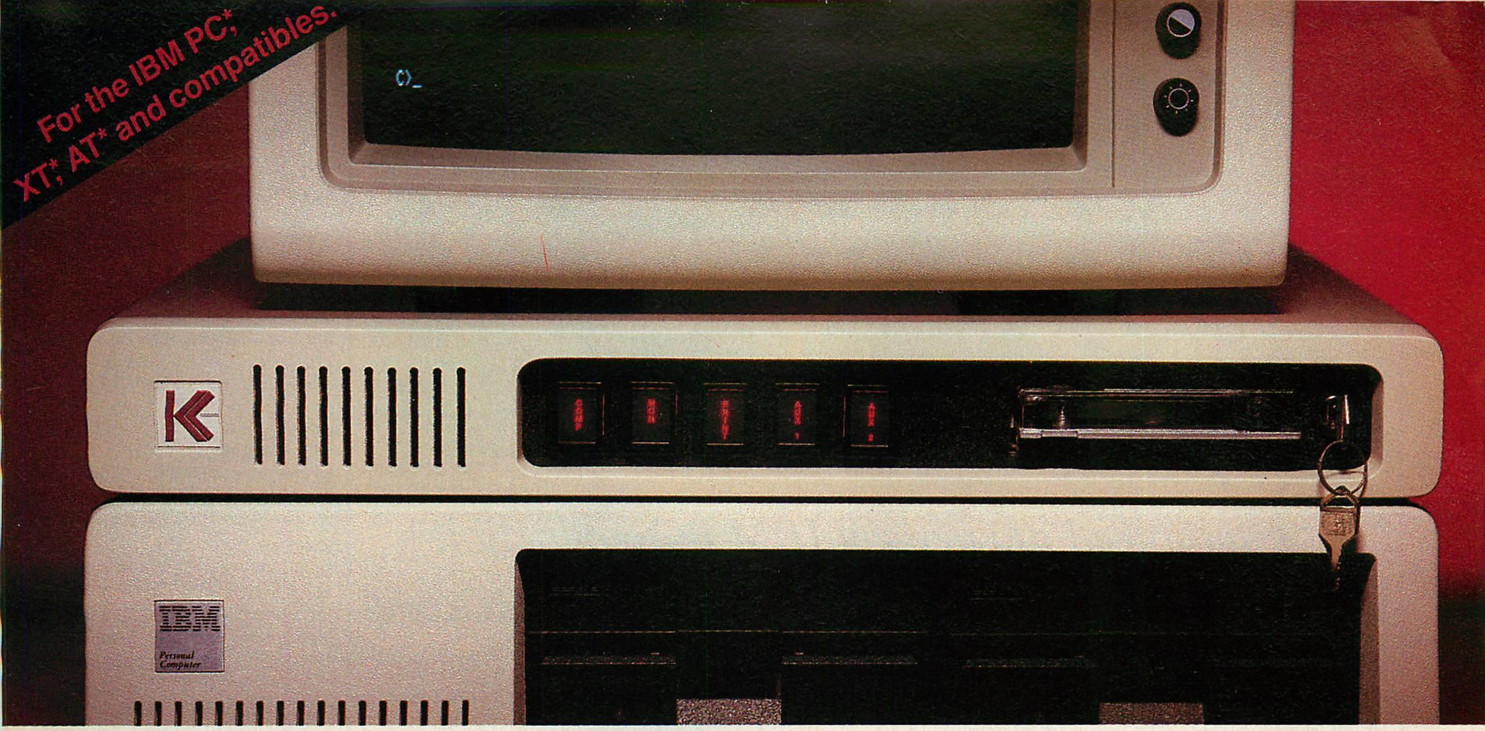
I am surprised and alarmed to learn that *PC Tech Journal's* regular department, Product of the Month, has been viewed with skepticism by many of our readers. We have been asked many times (too many times, in fact) if our write-up was an honest assessment or just hype designed to lure advertisers. Simply put: it's for real.

Perhaps an explanation about our selection process will help clarify our position. To begin with, the featured product is selected by the *PC Tech Journal* editorial staff; our decision is based solely on product quality and has nothing to do with advertising. Second, we consider only production copies of products, strictly in accordance with our standard review policy. Third, products under consideration are given full tests in our offices. We look for quality, value, design, and utility; we recently rejected a very fine product because of a small but annoying bug and will not consider the product again until a new release cures the problem. Finally, we write enthusiastically about these products because we believe their excellence deserves such mention.

We examine five to ten products each month. Neither the vendors nor our advertising staff knows which products are being considered. Vendors are usually notified of their product's selection at about the same time that the magazine appears in your mailbox; obviously, this is long after the evaluation is completed.

Product of the Month is designed to attract the reader's attention to significant products. It will continue to reflect the enthusiasm we feel for the featured products. Nevertheless, you can rest assured that our objectivity remains intact and that our selection process is always demanding.

—*WF*



We put hard disk storage and backup where it belongs.

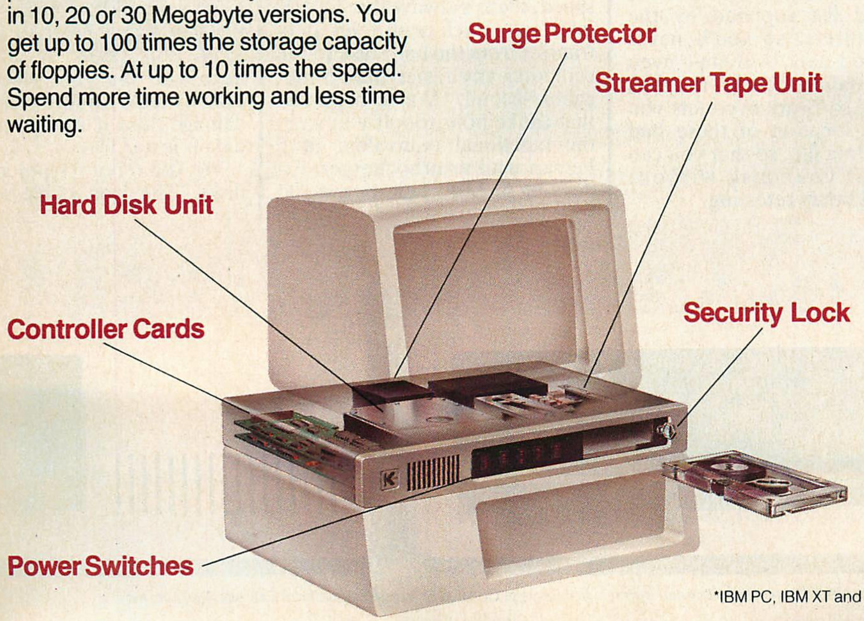
Maximum storage in a minimum space.
A mass storage solution for personal computers need not occupy massive amounts of space.

That's why Kamerman Labs packs an entire hard disk storage and backup system into a package only 2½ inches high.

In fact, our Masterflight system is so well integrated with your PC, it looks like it should have been there all along.

Hard disk storage.
Inside the Masterflight is a high-performance hard disk system, available in 10, 20 or 30 Megabyte versions. You get up to 100 times the storage capacity of floppies. At up to 10 times the speed. Spend more time working and less time waiting.

Streamer tape backup.
Any professional working environment demands a reliable backup solution. The Masterflight provides it through a high-performance streamer tape that backs up an entire 10 Megabytes in less than 4 minutes. Streamer tapes are available in 20, 40 and 60 Megabyte capacities. Instead of just a hard disk, Masterflight gives you a complete data storage solution.



Power control.
Masterflight directs power to your computer, monitor, printer and two other devices through front panel switches. No more blind groping on back panels to power up your system.

Surge protection.
A little power surge can produce a lot of damage. Masterflight protects your system with special circuitry that filters out spikes, surges and line noise.

Security.
Masterflight guards access to your system through a fully lockable power switch.

Phone today. Dealer and OEM queries welcome. 800-522-2237
In Oregon, Hawaii and Alaska call 503-626-6877

Kamerman Labs

8054 SW Nimbus, Bldg. 6
Beaverton, OR 97005
Phone 503-626-6877

Low power

Complete hard disk kit $395

Comes complete with virtually the identical controller that's in the IBM®XT, and Xerox® warranties the hard disk for one year

Guaranteed 100 percent IBM PC compatible

How can we offer this fantastic price? Simple. We buy in such volume that even the most avaricious hard disk businessmen understand they have to give us the best price possible. We could pocket the difference, but we don't.

Instead, we put the extra profit into our testing facilities. That's why Xerox guarantees our $395 10 megabyte hard disk for one year.

Xerox knows, as our customers know, that we have an extensive testing program. Here is what we contribute toward giving you the maximum hard disk performance.

Best Drives Available

First, we buy the best drives available. Sounds trite, doesn't it? I mean, a drive's a drive—right? Hardly. You should see some of the junk we get in our labs. Some have such high failure rates that we even questioned our own $10,000 hard disk tester. But when we tested other manufacturers' drives we were assured that our equipment was fine, which just confirmed that the bad hard disks were not only bad—they were real bad.

But that's just the weeding out process. We then take each drive that we've put through our tester and test it again with the controller you've requested. We call this a "tested pair."

DOS Doesn't Do It

In case you're thinking that all this is an unnecessary duplication of what DOS does for you, let me explain the disk facts of life.

If DOS did what you may think it is supposed to do when you format the disk, DOS would map around these bad areas. Unfortunately, DOS doesn't do this.

DOS 2.0 and 2.1 can't enter the bad tracks. DOS 3.0 can, but only on the IBM AT. Unfortunately, as the press has so well documentated, the AT's hard disk develops bad tracks later on.

We do what DOS can't

We believe the problem is so bad, we use a software program that performs a powerful test of your disk drive on all of the IBM or IBM compatible computers—PCs, XTs, and ATs. Our format takes hours to analyze the disk. But when we finish, you know that the bad tracks are really mapped out so you won't write good data that will disappear into a black hole. We even send you a printed statement of our test results.

Our software allows you to type in the bad track locations from the list supplied by the manufacturers, so you'll never write good data to them—even if DOS didn't identify them as bad. The software even lets you save the location of these bad sections to a file, so that you can reformat your disk without spending hours retesting.

We even include a program that will give you continuous comments on the status of your hard disk. No more waiting for that catastrophic failure.

Average Access Time

As you might suspect, some hard disks are faster than others in their ability to move from one track of data to another. The time it takes the hard disk to move one-half way between the beginning of the disk to the end is called the "average access time."

The first generation of 10 megabyte hard disks had average access times of 80-85 milliseconds (msec). But computer users love speed, and guess what—the average access time for the new 20 megabyte hard disk in the IBM AT is only 40 msec. (We sell an AT equivalent with only 30 msec access time!)

There are some legitimate reasons for the shorter access time. It's particularly helpful when there are multiple users on the same hard disk. It's also important when running a compiler. But remember, before you get too wrapped up in the access speed, there's always that ST 506 interface which won't let data transfer from the hard disk to the computer any faster than 5 megabits/second. We've bypassed that choke hole, too. If you want the functional equivalent of a Ferrari with a turbocharger, order our 10 Mbit per second 100 megabyte hard disk with 18 msec of average access speed.

Compatibility

To be sure that your hard disk is 100 percent compatible with the IBM XT you don't need to buy the same hard disk that's in the XT. You can't even be sure what brand hard disk it is because IBM, like Express Systems, goes into the marketplace and buys hard disks from several vendors. However, they buy their XT hard disk controller from only one vendor—the same one we do.

You can buy the IBM XT controller from IBM for $495 or you can buy from us, the functional equivalent, manufactured by the same company that makes it for IBM for only $195. Is it the exactly identical IBM XT controller? No, it's better. First, it takes less power, and secondly, it can control from 5 to 32 megabytes—the IBM controller can work with only 10 megabytes. It is 100 percent *IBM XT* compatible, and 100 percent is 100 percent. If you want to save a slot, we carry a version that lets you operate two hard disks and two floppy disk drives.

More than 32 Megabytes

You can operate with more than 32 megabytes (the limit of DOS) through the use of "device drivers." Express Systems can supply you with device drivers for our hard disks for over 32 megabytes formatted. But, if you don't have individual files, or databases that are large, you might want to consider one of our controllers that can divide our 65 megabyate (formatted) hard disk into two equal volumes of 32 megabytes each.

Reliability

We offer you a choice between iron oxide and plated media—the stuff that covers the hard disk and gives it its magnetic properties. Iron oxide is,—well, it's rust. If you inadvertently joust your disk, you may cause the low flying head to dig out some iron oxide. A little rust flake can ruin your whole day. Plated media is more resistant to damage, and if it happens, less data is lost.

We offer both types of hard disks. The iron oxide is older

technology, and quite frankly, manufacturers understand it better. Their better understanding, combined with some of the special head locking mechanisms, gives us peace of mind when we sell you one.

Power

Hard disks consume power. Our small, half-high hard disks consume so little power that you can use them with your existing IBM PC power supply. If you plan to use lots of slots, you'll want to increase your power supply to be safe. We offer the same amount of power for your PC that comes in the XT.

Our Customers

Some folks just never feel comfortable buying mail order. They forget that Sears began as a mail order house or that IBM is now into mail order. But, if it helps, here is a *partial* list of customers who have felt comfortable to buy from us.

IBM	Sears
American Express	Honeywell
U.S. Army	MIT
AT&T (Bell Labs)	RCA
Bausch & Lomb	Lockheed
Xerox	Sperry

Easy to Install

If you're like most of us, raised on the boob tube rather than the Great Books, you'd rather see the movie than read the book. Well, now you can choose to read our installation manual or for only $9.95 more, you can get a VHS or Beta video cassette showing the simple steps for installation.

Warranty

We offer you a one year warranty on our hard disks—the same as IBM on the AT and 90 days on the tape drives. (It's all the manufacturer gives us.) If anything goes wrong with your tape or disk drive or hard disk, send it back in the box it came in. However, we have found that we can usually solve the problem over the phone. So call first for a return authorization number because we can't accept any returns without it.

Complete Hard Disk Kits

Formatted MB	Height	Plated Media	Average Access	Transfer Rate	PC or PC/XT	AT
10	½	no	85 msec	5 Mbits/s	$ 395	$ N/A
10	½	yes	85 msec	5 Mbits/s	$ 495	$ N/A
21	½	yes	85 msec	5 Mbits/s	$ 795	$ 595
21	Full	no	30 msec	5 Mbits/s	$ 1,535	$ 1,340
32	½	yes	85 msec	5 Mbits/s	$ 995	$ 795
32	Full	no	30 msec	5 Mbits/s	$ 1,775	$ 1,575
65	Full	no	30 msec	5 Mbits/s	$ 2,295	$ 2,070
100	Full	yes	18 msec	10 Mbits/s	$ 4,995	$ 4,995
Removable Hard Disk						
10	½	no	90 msec	5 Mbits/s	$ 1,095	N/A

Tape Systems and Subsystems

Formatted Storage Capacity	Height	Data Transfer Rate (k/sec)	PC or PC/XT	AT
60 Mbytes	½	88	$ 995	$ 995
60 Mbytes Subsystem		88	$ 1,295	$ 1,295
21 Mbytes (unformatted) Start/stop Subsystem		24	$ 595	$ 595
26 Mbytes Floppy Tape® Subsystem		31	$ 749	$ 749

Controllers

All of our hard disk and tape controllers are available separately: Please call for prices.

Subsystem Chassis

Any of our disk or tape units are available in an external subsystem for an additional $250.00. You can mix & match any of our ½ high hard disks or tape drives together or add any single full height hard disk.

Tape Cartridges

Express Certified 555 foot 310 Hci ¼-inch Data Cartridge $35.00

Power Supply

130 Watt Power supply $75.00*
150 Watt Power supply $125.00

*with the purchase of any drive

Comes complete

All Express Systems products come complete with the appropriate software, tape and/or hard disk controllers, and cables where required. Hard disks are formatted and tested with the PC DOS of your choice. All drive sizes are formatted capacities.

If your application requires a stacking kit, power splitter cables, daisy chain cable, or some other variation, we'll supply these items at a nominal charge. We even ship our hard disks with Command Assist™ an on-line DOS-like manual to give you help with your DOS commands.

More questions?

Because we spend so much attention on the front end with ensuring that our disks will arrive in working order, we have a customer service department that, unlike many of our competitors, has little to do. When you need us, you won't get a constant busy signal.

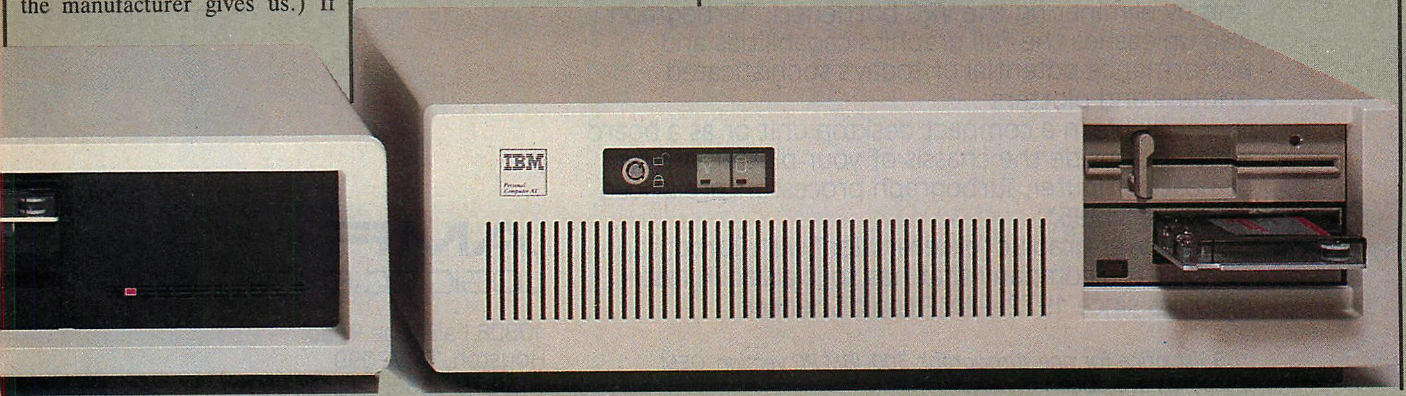
Call our friendly, knowledgeable customer service staff to get answers to your questions—before or after the sale. Our people, who know the PC, can talk you through the sticky parts, and they'll respond to you quickly. Just call us.

How to order

Pick up the telephone and call 1-800-341-7549, to order. We accept Master Card, VISA, American Express and Diners Club. Or send a cashier's check or money order (We'll take a check, but you'll have to wait for it to clear) and tell us if you want one of our recommended configurations or you want to mix and match yourself. Corporations with a DUNS number may send purchase orders for quantities over five.

EXPRESS SYSTEMS

Call Toll Free 1-800-341-7549 Ext. 1300
In Illinois call (312) 882-7733 Ext. 1300
Express Systems, Inc., 1254 Remington, Schaumburg, IL 60195



Command Assist is a trademark of Micro Design International

CIRCLE NO. 155 ON READER SERVICE CARD

The graphic excellerator. Turbograph.™

Accelerated performance. Excellent graphics. At an exceptionally low price. That's Turbograph—the innovative line of high performance graphics processors from AMF Logic Sciences.

Turbograph responds to the soaring demand for improving and streamlining computer-generated graphics in all sectors of the market. From IBM PCs creating business graphics to VAX-type systems in CAD/CAM environments.

Turbograph accelerates the time it normally takes to process and convert graphics from vector to raster form. It relieves the host computer from the costly overhead of vector-to-raster conversion. And by eliminating the VRC bottleneck, Turbograph also unleashes the full graphics capabilities and performance potential of today's sophisticated printers and plotters.

Available in a compact desktop unit or as a board which fits inside the chassis of your plotter, printer, or host computer, Turbograph processors are priced as low as $1995.*

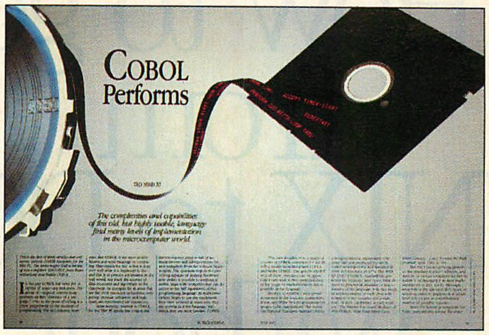
Discover graphic "excelleration" with Turbograph. Contact the company that has specialized in raster graphics since 1972—AMF Logic Sciences.

**U.S. list price for one Turbograph 300, IBM PC version; OEM discounts available.*

AMF
LOGIC SCIENCES, INC.

10808 Fallstone Road
Houston, TX 77099
713/879-0536; telex 706691

CIRCLE NO. 131 ON READER SERVICE CARD



SLOW PERFORMING PGD

Thomas Hoffmann's review of IBM's Professional Graphics Controller and Display in the July 1985 issue ("Power Graphics," p. 56) provided your readers with a good examination of the graphics capabilities of the adapter and monitor; however, the review only cursorily examined its performance in other areas. As noted in the article, due to slow performance in text mode, it was recommended that a monochrome monitor "would be a valuable addition" to a system. There is another problem with the display that is related to its slow performance in text mode. The Professional Graphics Display is so slow that it cannot keep up with communications through the serial asynchronous adapter at speeds in excess of about 600 baud. It consistently loses characters. Thus, for 1200- and 2400-baud modem users, the PGD is certainly deficient. When notified of this problem, IBM's response was *caveat emptor*.

John M. Scott Bryan, Ph.D.
Norman, OK

REALIA EVALUATION

I object strongly to your placement of Realia COBOL on the "Not Recommended" list of COBOL compilers ("COBOL Performs," Ted Mirecki, August 1985, p. 107) because it is copy-protected. It should have been rated right behind Micro Focus Professional COBOL (or ahead of it based on price/performance) with a caveat about the copy-protection scheme. We at The Coca-Cola Company have used this compiler heavily for more than a year and have experienced no problems with the copy-protection.

By not installing the product on hard disks, the potential problems with that procedure are avoided. The SuperLok scheme allows one back-up to be made and stored in a safe place. These points negate both of Mr. Mirecki's criticisms. All that remains is the minor annoyance of placing the master disk in drive A: to start the compiler. This is a small price to pay for a product of such potential.

It seems Realia has been placed on the sacrificial altar in the name of unprotected software. If Mr. Mirecki wanted to strike a blow against copy-protection, it should not have been in a comparative product evaluation at the expense of one manufacturer. By allowing this practice, you compromise the integrity of your evaluations.

John A. Lock
Atlanta, GA

We stand by the evaluation. PC Tech Journal *agrees with Mr. Mirecki's conclusion that copy-protection is anathema for software tools; we go further and condemn the current software techniques for all programs because of the many problems (notably hardware failure and back-up problems) they induce. (See Directions, "User's Rights," September 1985, p. 9.)*

Realia is cited in the article as strongly recommended otherwise.

—WF

ENHANCED COLOR

Thank you for the informative article on the IBM Enhanced Graphics Adapter in the April 1985 issue ("Graphic Enhancement," Thomas V. Hoffmann, p. 58). Is it possible to set the overscan color while in the enhanced mode to obtain a border color other than black and, if so, how? Efforts using function 10H of INT 10H or I/O port 3C0.11 produce only an outline around the main background area of my Enhanced Color Display.

Could you also comment on the *Technical Reference Manual* referred to in the article? I have been unsuccessful in my attempts to obtain any literature from IBM covering the hardware or the ROM BIOS listings.

Walter A. Puryear
Athens, GA

I was so impressed by the evaluation given the EGA in the April 1985 issue that I dashed out and bought it and the Enhanced Color Display. Unfortunately, I am unable to determine from the tables in your excellent article how to set the border color when the EGA is in enhanced mode (with switch 1 off). I can do it in CGA mode (with switch 1 on) using the normal BIOS function call 10H with AH=0BH, BH=0, and BL=1. However, this sequence has no visible effect in EGA mode. I do not understand the notation used in sidebar 1 to refer to the new palette registers and the attribute controller: what does 3C0.00-0F mean? 3C0.11 [0-5] in the section on overscan color?

If you could provide me with an assembler coding example for setting the overscan color, I could probably use that as a model for everything else. If you could tell me how to obtain a copy of the EGA ROM BIOS listing I would be most appreciative, since the dealer I purchased my system from decided I did not need the new diagnostic diskette, the inserts for the various manuals, or the installation instructions, and promptly threw them in the trash after assembling the system. For this reason I am unsure whether a copy of the listing was included with the card or not. Even the two-volume hardware technical reference manual that I have inspected does not cover the EGA. I am quite willing to pay for a copy if only I knew where I might obtain one.

I am using SideKick, TopView, Displaywrite 3, WordStar 3.3, Lotus 1-2-3, and BASICA 3.0. Everything works well on the EGA except TopView, which seems to have trouble getting the little blinking blue bullet to align itself over the little black bullet. The cursor acts strange in all the programs run under TopView: it blinks in the middle of the line, not at the bottom. The colors are superb compared to the normal color display, as is the text.

All things considered I believe we made an excellent choice for the future. Your article played a major role in that decision, for which you have my thanks.

Ray Smith
Costa Mesa, CA

The notation port.index-index *indicates multiple device registers addressed through a single I/O port address. For example, the 16 palette reg isters in the attribute controller chip of the EGA are indicated by 3C0.00-FF. To set palette register 3 (3C0.3), you must write a 3 to bits 0-5 of port 3C0 to set the internal register address, then write the color value to port 3C0. The brackets indicate a bit field; thus, 3C0.3[0-5] means the low-order six bits of internal register 3 accessed through port 3C0. Different devices may use different ports for setting the internal register address.*

The border color can be set through 3C0.11[0-5], but in the high-resolution, 350-line modes produces the thin outline observed by Mr. Puryear. This is because the CRT timing parameters for these modes define a larger blanking interval that extends into the visible area of the screen and a correspondingly small border area. You could experiment with different CRTC set-ups, but I doubt you would have much luck. I have resigned myself to black borders in the EGA high-resolution modes, but am not happy about it.

The technical reference documents for the EGA, Professional Graphics Controller, and other new devices are supposed to be released as updates to the two-volume Options and Adapters Technical Reference. Obviously, IBM is late in making these important materials available to advanced users. The company provided preliminary copies of the technical documents with our evaluation units. Unfortunately, IBM is the only game in town for these documents—perhaps letters to the president might help to dislodge them.

—Thomas V. Hoffmann

MINUS THE OVERHEAD

I enjoyed William Hunt's article on C libraries in the June 1985 issue of *PC Tech Journal* ("Drop-in Modules For C," p. 100). However, I felt that there was some important information omitted, at least in the dot-testing function.

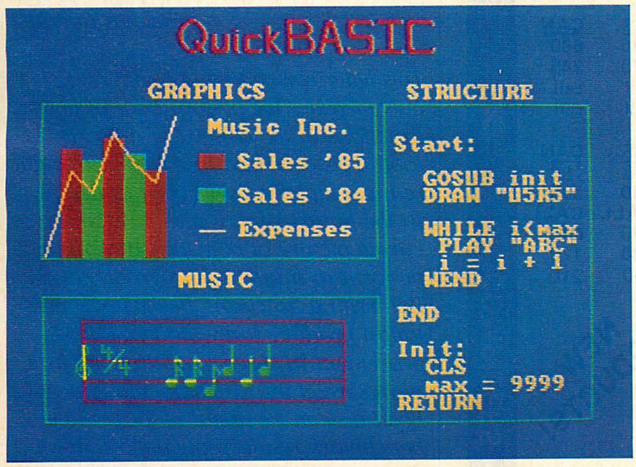
No reference was made to the time spent in dot.c itself. It seems the review assumes that the time spent in the testing routine for dot.c in the loop is insignificant when compared with the time spent writing the dots. I checked

Get up to speed for under a hundred bucks.

If you're writing in the BASIC that came with your IBM® PC, now you can make your programs scream. Microsoft's new QuickBASIC Compiler will punch you right into warp speed, accelerating your programs by as much as ten times. And all with hardly any code changes.

You get all this at a price that won't slow you down, either. Just $99 and you're in.

The new QuickBASIC Compiler was designed to work with the



programs you've already written. It supports all BASIC sound and graphic statements including PLAY, SOUND, LOCATE, DRAW, GET, PUT, LINE, CIRCLE, PSET, and COLOR.

So your graphics and music routines will look and sound as they should.

And when you're ready to move into structured programming, QuickBASIC also offers language extensions for a speedy transition. With optional alphanumeric labels you can make your programs more readable. And compiled subprograms will save you time.

So get moving. Call us now for more information on the QuickBASIC Compiler and the name of your nearest Microsoft dealer. Just ring (800) 426-9400. In Washington State, Alaska, Hawaii and Canada, call (206) 828-8088.

MICROSOFT®
The High Performance Software™

Then go give your programs a swift kick.

Microsoft is a registered trademark and The High Performance Software is a trademark of Microsoft Corporation.
IBM is a registered trademark of International Business Machines Corporation.

the time of the routine without writing dots, using the Lattice C compiler on a PC running at 4.77 mH, and came up with a time of 0.05 seconds. It seems that the testing routine accounts for more than 25 percent of the time spent in the whole routine by the C Utility Library. If you subtract the overhead from each test, the C Utility Library routine looks even better: instead of being only a little better than twice as fast as its nearest competitor, it is almost three times as fast.

Also, I would dispute Mr. Hunt's inference that Halo provides significantly faster screen graphics for dot and line than the C Utility Library. I have written a routine to write a dot directly to screen RAM, and can testify as to the limits of the software. From looking at the testing routine, I would bet that C Utility's line-drawing speed is even more impressive than the data show. I cannot be accused of bias; my company publishes a package that competes with the C Utility Library.

Perhaps other readers have checked the other testing routines with respect to their impact on the data. Their input would be valuable if they found other places where the testing routines' times were significant.

By the way, 0x3f and 0x7f are decimal 63 and 127, respectively, referring to the row and column in dot.c.

All in all, the article is a very useful reference for those considering the purchase of C libraries.

Tom Hogan
C Source Inc.
Kansas City, MO

This letter raises a good point. All my tests measure the performance of library functions in the context of a simple loop. This overhead in the measurement reduces differences between functions in the libraries but it also gives an accurate picture of how the functions will perform in an actual program. For the dot-writing functions, better implementation produces only moderate improvement in speed. Line-drawing functions spread the function-calling overhead over more useful work, and the benchmark results show important differences for the libraries tested.

Some useful lessons can be learned here. No general purpose dot-writing function will be fast enough for writing whole screens. Better implementations of dot-writing functions produce only moderate-improvements in performance. But well-implemented functions to draw lines, circles, or other figures or

Pascal Unlimited.

Once you've gotten some experience under your belt, you'll find that even with a very decent program like Turbo Pascal,® you start scraping the ceiling of its abilities. You need a Pascal that's conditioned for the serious developer.

New Microsoft® Pascal.

A totally comprehensive and complete package, it lets you tackle large projects by giving you the utilities, features, and support libraries that professionals expect.

Unlike Turbo, Microsoft Pascal offers separate compilation. So you can create your own module to use and re-use in your Microsoft language programs.

In addition to full XENIX® compatibility, our compiler also supports the new Microsoft and IBM® Networks.

If you've already invested in Turbo Pascal, we'll make it worth your while to move up. We're offering a rebate of $40 when you upgrade to Microsoft Pascal.

MICROSOFT.
The High Performance Software™

For the name of your nearest Microsoft dealer, call (800) 426-9400. In Washington State, Alaska, Hawaii and Canada, call (206) 828-8088.

Microsoft Pascal Compiler Version 3.3 for MS-DOS

	Microsoft Pascal v 3.3	Borland Int'l Turbo Pascal v 3.0
Separate module compilation	Yes	No
Conditional compilation	Yes	No
Full memory usage (to 1 megabyte)		
program code	Yes	No
data	Yes	Yes
Overlay support	Yes	Yes
Math library support		
8087/80287 emulation	Yes	No
8087/80287 coprocessor support	Yes	No*
Floating-point	Fast IEEE	non IEEE
BCD floating-point	Yes	No*
MS-DOS® 3.1 network support (incl. IBM LAN)	Yes	No
Link multiple routines	Yes	No
Link existing third-party libraries	Yes	No
Link with Microsoft FORTRAN, C and Macro Assembler	Yes	No
Relocatable object format	Yes	No
Transport source between MS-DOS and XENIX	Yes	No
Do source level debugging	Yes	No
LINKER included	Yes	No
Library Manager included	Yes	No
Utility to modify and examine header	Yes	No
Compress utility	Yes	No

Pascal Benchmarks—done on a COMPAQ Plus™ with 512K memory with no 8087

	—Execution Time—	
Gauss-Seidel	:05.15	:07.60
Sieve of eratosthenes	:13.15	:15.88
Trig	:13.11	:34.97

*Option available separately

to fill areas inside figures, can provide significant improvements in speed. The mark of a good graphics library is that it provides powerful primitives, efficiently implemented so that it is not necessary for the application to write single dots on the screen.

—William Hunt

SNA SEMANTICS

I read with interest the article by Art Krumrey entitled "SNA Strategies" in the July 1985 issue of *PC Tech Journal* (p. 40). However, I feel I must point out some inaccuracies that detract from what is otherwise a fine piece on future SNA directions in the PC arena.

Leaving aside the allegation that SNA is characterized by a "cable-and-controller infrastructure," I question whether the PC's "coming of age" will enable SNA to "realize its full potential" as claimed in the article. This is rather like having $999,999.75 in the bank and claiming that the next quarter will make you a millionaire. It may be literally true, but that next quarter is unlikely to affect anyone's activities.

But the real fault of the article is in the area of Mr. Krumrey's claims regarding the relationship of SNA to the International Organization for Standardization (ISO) Reference Model for Open System Interconnection. This has been an area of research for me for the past several years, and I am very sensitive to anyone's attempts to make the ISO/OSI model a shibboleth among computer manufacturers.

Let me state unequivocally that the preconceived relationship between the Open Systems Interconnection (OSI) subcommittee of the International Standards Organization (known as OSI/TC97/SC16) and the IBM networking solution known as System Network Architecture (SNA) is exactly nil.

This is not to say that there is *nothing* in SNA that corresponds to the ISO/OSI model. It does mean that existing correspondence is merely accidental, or generic to any networking strategy. There is a simple (but not exclusive) reason for this: SNA dates from 1974; the ISO subcommittee was formed in 1978. Therefore, networks predating 1978 must have been completely restructured to conform to the model. IBM has not done so with SNA; it cannot without sacrificing many old but still supported features of SNA. Many users are thankful that IBM is not willing to do so merely to satisfy a standard.

IBM has never officially (to my knowledge) *claimed* that SNA corresponds particularly well to the ISO/OSI model. But certain journalists and professors seem to enjoy plugging SNA into a structure in which it has no business being. IBM's position vis-a-vis the ISO/OSI model is the same regarding other such de facto standards as EBCDIC vs. ASCII: just as IBM has felt no compulsion to build ASCII computers, it has felt no compulsion to fit SNA into the ISO/OSI model (the term is not even mentioned in the IBM System Network Architecture Technical Overview).

I will not attempt a detailed analysis of Mr. Krumrey's presentation of the "seven levels of SNA" for the sake of brevity. Suffice it to say that IBM's own training programs present SNA as three layers, not seven. IBM refers only to Application, Function Management, and Transmission Subsystem. Also, an attempt to characterize CICS or TSO as valid implementations of the Presentation Level is an abuse of the term.

This letter is not intended as an attack on IBM, SNA, or Mr. Krumrey. I

Non-stop flights to XENIX.

The XENIX® market is taking off. To keep up you've got to convert your MS-DOS® software to run on the new, faster 80286 machines. Quickly. Efficiently.

Nobody makes porting software to the XENIX environment as easy as Microsoft; you simply recompile the original source code.

Our full-featured XENIX languages use commands and syntax identical to their MS-DOS siblings. So you don't have to rewrite your program to move it.

Whether your applications are written in Microsoft versions of Pascal, BASIC, COBOL or FORTRAN, we have the solution to your transportation problems.

MICROSOFT®
The High Performance Software™

For the name of your nearest Microsoft dealer call (800) 426-9400. In Washington State, Alaska, Hawaii and Canada, call (206) 828-8088.

Overall features of XENIX Languages
- Shell support of syncronous and asyncronous tasks.
- Support of pipes for interprocess communication as a generalized I/O device.
- Record and file locking:
  – For sequential, random, dynamic and ISAM files.
  – Blocked or unblocked file reads.
  – Deny access, deny read and deny write lock levels.
  – Automatic or manual record and file locking of ISAM files.

Microsoft FORTRAN Compiler
- Based on ANSI 77 (x3.9-1978) standard.
- Single-and double-precision for both real and complex data types.
- Math packages: 287 support, alternative fast math and BCD decimal math.
- Large program support.

Microsoft COBOL Compiler
- High Level 2 ANSI 74 standard features.
- Multi-key ISAM with split keys, alternate keys and duplicate keys.
- Interactive extended screen section.

Microsoft BASIC Interpreter
- Standard Microsoft BASIC.
- Supports trace, single step and immediate execution.
- Supports random, dynamic, ISAM and sequential file I/O.

Microsoft Pascal Compiler
- ANSI and ISO standard with extensions.
- Math packages: 287 support, alternative fast math and BCD decimal math.
- Large program support.

have merely sought to correct some of the misleading impressions the article gives about SNA and the ISO model.

Walter J. Goralski
Elmsford, NY

I appreciate your remarks. We have, in all other networking articles, described the architecture in terms of the ISO/OSI model. This is one of those unfortunate situations in which IBM articulates its position differently than the rest of the world, and we are caught in the middle, trying to explain perhaps unfamiliar SNA concepts to an audience more exposed to ISO terminology (both by us and by network vendors).

—WF

THE SMART WAY

I found "Exceeding the Speed Limit" (Tom Puckett, June 1985, p. 86) both interesting and well-written. I particularly appreciated the inclusion of the basis for the benchmark test—the specifics of the formula used and the spreadsheet array into which it was placed. This permitted me to check the accelerator board ($1,900-$1,400) performance against the software I use (Innovative's SMART Series, $600) and against the 8087 numeric coprocessor ($150).

Using the most recent version of SMART, I duplicated the formula (*ln(sqrt(x**1.01)+1)) and its rolling configuration (where *x* is picked up from the left adjacent cell, the first cell in each row picks up the last cell in the preceding row, and the first cell in the first row picks its value from the cell immediately above it, which contains a number but no formula), in the same ten-column-by-100-row array you used.

The results were as follows: Baseline (article), 48 seconds; Accelerator board (article), 20 seconds; SMART, 28 seconds; SMART and 8087, 4 seconds.

Although I do not think the intent of the article was to publish a definitive exploration of options, it appears to me the best (read "cheapest") way to accelerate your spreadsheets is to buy a $600 software package and a $150 coprocessor instead of a $1,400-$1,900 board.

Terrance G. Logan
Lynchburg, VA

Agreed. That is one of the problems we all encounter when building a system configuration—there are just so many combinations and possibilities. I wish we had the time and resources to check them all, but we are always glad to hear about useful alternatives.

—WF

POPDOS2 TESTIMONIAL

John Walkenbach's gymnastics (Tech Notebook 45, "Printer Control," August 1985, p. 41) to achieve printer control from SideKick or Filer, without exiting to DOS level, convince me I made the right choice in going for Bellsoft's Pop-Up DeskSet. I invoked PopDOS2 with Alt+U while using WordStar to compose this letter. F10 selects the Print Menu, which includes the SET submenu (via F4). SET allows me to send Form Feed, Line Feed, and 12 print control characters to the IBM Graphics Printer. To send control characters to nonstandard printers, I use CODE submenu via F5, which allows me to send any sequence of ASCII codes by entering three-digit (decimal) numbers.

Bellsoft's DeskSet also contains modules for Alarm Clock, Calendar, Notepad, Clipboard, Calculator, and telecoms. Your comparisons with SideKick and Filer would be useful.

Stan Kelly-Bootle
Mill Valley, CA

NONLINEAR CLARIFICATION

The article "Nonlinear Least-Squares Fitting" by Walter Schreiner, Michael Kramer, Simon Krischer, and Yedidyah Langsam (May 1985, p. 170) lists a very useful program containing a number of nice features. The user interface is very well thought out making the program easy to use. The use of the Cholesky square root method to solve the normal equations is not only efficient but stable and accurate, unlike some pivot algorithms. At the time this article appeared, we were faced with the problem of evaluating Nuclear Magnetic Resonance spin lattice relaxation times from a fit to a nonlinear three-parameter function. In the equation below, PARM(3) is the desired parameter, and we obtained a good value with little trouble.

We did find, however, two errors in the program that could give considerable difficulty. The program contains two subroutines (at lines 20000 and 21000) that define the fitting function and the derivatives of this function with respect to each parameter. The first problem is that subroutine 21000 calls 20000 and assigns the resulting value of FUNCTN to the variable F. It appears as though F is used only in subroutine 21000 to simplify the calculation of the derivatives in the example function for the game scores. But subroutine 21000 is called also at line 6380 where both the function and the derivatives are needed in the next several lines of code. Statement 6390 uses F for the function value, not FUNCTN. The variable F is used again in another part of the program (lines 5060-5180), and failure to calculate F as well as FUNCTN in subroutine 21000 has catastrophic results. A better strategy would call subroutine 20000 before statement 6390 and consistently use FUNCTN instead of F in 6390. This is done in other parts of the program where FUNCTN is needed.

A second error is more subtle and thus harder to find because it does not show up in fits where fewer than three parameters are adjustable. When the program was used to fit the function:

```
FUNCTN = PARM(1)*(1 -
  PARM(2)*EXP(-X(1)/PARM(3)))
```

the errors in PARM(2) and PARM(3) were, to seven digits, identical, even though the values of PARM(2) and PARM(3) differed by two orders of magnitude for the data used. Indeed, when a four-parameter fit was tried, the errors for PARM(2), PARM(3), and PARM(4) were exactly the same. The errors in the parameters (stored in DPARM(14)) result from calculation in lines 7090-7120. The problem is that the index *K* used for *G(K)* is changed in subroutine 4500.

It turns out that the value of *K* returned from subroutine 4500 will always be 1 (see lines 4670-4750) so that DPARMs calculated for *G(K)*'s having *K* greater than or equal to 2 will always be the same. The solution to the problem is to change the index *K* in lines 7090-7110 to, say, *K4*.

Both of the problems mentioned demonstrate one of the inherent limitations of Microsoft BASIC when used to write large programs—the global nature of variable names. Both coding problems involve the same variable names used in parts of the program separated by many dozens of lines. Were external subroutines with parameters and local variables possible with this dialect, the problems mentioned above would be less likely to occur.

Armand A. Fannin, Jr., Lt. Col., USAF
Richard D. Bertrand
Colorado Springs, CO

The first "error" mentioned by Lt. Col. Fannin and Mr. Bertrand would occur only if lines 21010-21020 are left out when the derivative subroutine is coded. Unfortunately, we did not point out that these lines are always required. *We do prefer the strategy suggested by Fannin and Bertrand, because it leads to less confusion; the structure that we had coded was a carry-over from the program's FORTRAN days. The second error mentioned is indeed a real bug, and our thanks to Fannin and Bertrand for their diligence in finding it. All occurrences of the variable K in lines 7090-7110 should be changed to K4.*

In preparing this article, our contention was that it is beneficial to adapt useful programs to the microcomputer BASIC environment, the deficiencies in the language notwithstanding. Interpreter BASIC affords the rare luxuries of easily modified code for adapting to special situations that arise, a good debugging environment for tracing elusive problems, and a compiler for pouring on the speed. Accordingly, we had included a section in our original manuscript on the pitfalls of translating a program from FORTRAN to BASIC, which was edited out of the final article for space considerations. To excerpt:

> *In FORTRAN there is no communication between the variables in the main program and those in subroutines, except via the arguments passed to the*

THE MOST POWERFUL, MOST COMPLETE dBASE PROGRAM DEVELOPMENT TOOL EVER!

FLASH CODE™

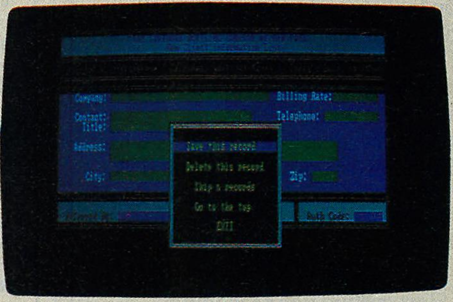
Program Developer for dBASE II & dBASE III

INTRODUCTORY PRICE $150.

Actually "DRAW" your dBASE screens!

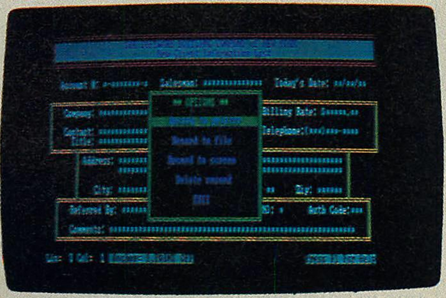
Take full advantage of all special characters, draw boxes and lines, use the color select menu, move pieces of the screen around, and more!

Specify variable names, file variables, data types, acceptable data ranges, pictures for edit checking, initial values, calculated fields, indexes, and more in a flash!



FLASH Code writes your program in either dBASE II or dBASE III!

Use it as is, modify it, or merge it with your own code. You can even use our new FLASHUP module to instantly flash up even the most complicated screen designs. No more waiting for slow print statements to display your screens.



POP-UP Windows

Create pop-up menus and pop-up help screens with our window editor. Use the FLASHUP module and have them pop up all over your dBASE applications.

NO RISK DEMO OFFER! TRY IT—YOU'LL USE IT!

Here's a no-risk offer. Order now and you'll also get a full demo disk. Use the demo and the manual for 30 days. If you don't love it, return the package for a full refund!

THE SOFTWARE BOTTLING COMPANY OF NEW YORK

Credit card orders only call 24 hours a day, 1-800-824-7888, operator 268.
For all other orders and inquiries call or write: The Software Bottling Company of New York, 6600 L.I. Expwy, Maspeth, NY 11378, (718) 458-3700. If we're shipping to a NYS address, please add 8¼% sales tax.

CIRCLE NO. 191 ON READER SERVICE CARD

> *subroutine, and variables that are specifically declared common to both. . . . BASIC, on the other hand, recognizes all variables as global whether they are in a subroutine or not. In order not to have our BASIC subroutine inadvertently change variables in the main program, each "local" variable in a subroutine had to be given a distinct name from any other variable in the program.*

Clearly, being aware of a pitfall does not make one immune to it.

We would love to see BASIC contain facilities for local variables and externally called subroutines; version 2.0 of the IBM BASIC compiler does provide for this. We hope IBM soon will match these new features of the compiler in the next release of interpreter BASIC.

—Walter Schreiner, Michael Kramer, Simon Krischer, Yedidya Langsam

IT NEEDED A PUSHA-POPA

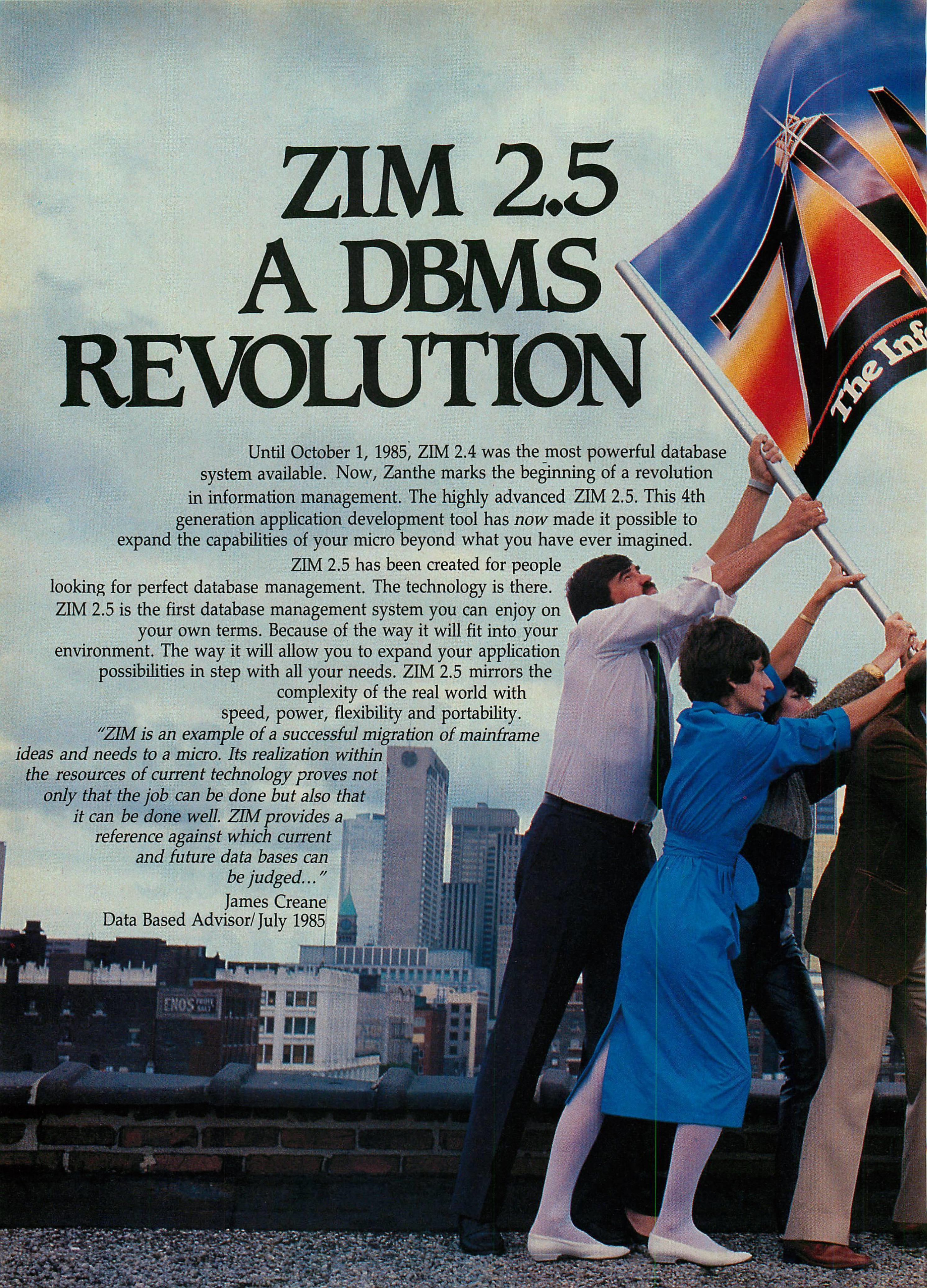
Tech Notebook #37 by Kevin Crenshaw ("Rev Up the AT Keyboard," May 1985, p. 39) was excellent. It saved me much research time to have a base program from which to work. With the elimination of the too small keyboard buffer by SuperKey and with the keyboard set as fast as it will go, the PC/AT really responds. I did have a problem, however, that other readers might also encounter. What I found was that different machines might require small changes to the code as shown in the magazine listing in order for the program to work.

Perhaps my 8042 controller is not as fast as those on Mr. Crenshaw's machines because I had to put some delay code after the XMTWT2 label. A PUSHA-POPA slowed down the checking of the controller enough to give it time to read the data byte. Without this code the program would not work. Also, the first character returned from the controller after the command byte was sent was not the ACK character (OFAH), but a 09CH. An ACK character was available on a second reading. I have not been able to determine just what a 09CH is, but if the check for the ACK character is removed the program works fine. The controller merely cares that you read something, not a specific something.

This program was very helpful because unlike many utility routines it solves a problem not addressed by commercially available and supported packages. Keep publishing those kind of tips in your excellent magazine.

George E. Defenbaugh Jr.
Tulsa, OK

On October 1, 1985 one product made all other DBMS obselete.


ZIM 2.5
A DBMS
REVOLUTION
Until October 1, 1985, ZIM 2.4 was the most powerful database system available. Now, Zanthe marks the beginning of a revolution in information management. The highly advanced ZIM 2.5. This 4th generation application development tool has *now* made it possible to expand the capabilities of your micro beyond what you have ever imagined.
ZIM 2.5 has been created for people looking for perfect database management. The technology is there. ZIM 2.5 is the first database management system you can enjoy on your own terms. Because of the way it will fit into your environment. The way it will allow you to expand your application possibilities in step with all your needs. ZIM 2.5 mirrors the complexity of the real world with speed, power, flexibility and portability.
"ZIM is an example of a successful migration of mainframe ideas and needs to a micro. Its realization within the resources of current technology proves not only that the job can be done but also that it can be done well. ZIM provides a reference against which current and future data bases can be judged..."
James Creane
Data Based Advisor/July 1985

PORTABILITY: With the multitude of hardware and operating systems available today, application portability is essential to software developers and large corporations. ZIM is available under PC-DOS, Concurrent PC-DOS, MS-DOS, UNIX, XENIX, and QNX. ZIM is the only database management system available for single-user and multi-user configurations with 100% application portability. Never again will you be required to re-write your applications for different operating systems environments. This will save you time and money.

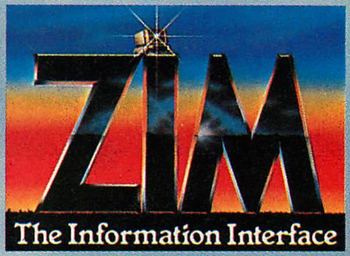
SPEED: *"ZIM has not sacrificed execution speed to gain its expanded functions in terms of existing microcomputer databases..."* James Creane, Data Based Advisor/July 1985. In comparison to industry leaders ZIM was at least 3 times and as much as 50 times faster in sorting and joining files within the database. Only ZIM can break the speed limit. ZIM's internal architecture, strategy selector, priority-driven buffering ability and implementation of a new and compact B*trees ensure that the information is delivered in the most efficient manner possible. Never again will sorting, reporting or other complex information retrievals be left as an overnight task.

POWER: *"ZIM is unique, and the source of its extraordinary power to mimic the real world springs from the way it views data. It uses an "Entity-Relationship" model of data, an extension of the relational model...(ZIM) represents and manipulates complex data with dramatic simplicity and economy of expression."* John Helliwell, PC Magazine/ September 3rd 1985.

With ZIM's high-level, natural language you can build user commands implementing applications with unparalleled ease and flexibility without the necessity for additional programming tools and additional costs. ZIM's forms facility permits completely menu driven applications. This, combined with the extensive report generator, allows development of applications that can be used by any novice user. The ZIM compiler will compile complete applications leading to faster execution and preventing unauthorized access or modification. Completely compiled applications can be run with the standalone Runtime system, decreasing cost and memory requirements.

FLEXIBILITY: *"ZIM provides a powerful interactive command language to invoke its information management functions...(it) is the first data base management I have tested to fully utilize...the edit command keys (F1, F2, F3)"*...Giovanni Perrone, PC Week/ May 7th 1985.

The only limits with ZIM are those of your hardware, operating system and imagination. Now with ZIM 2.5 your database is no longer limited to a single hard disk. With ZIM, information management is a breeze. With only three commands required for all update operations. ZIM gives you unprecedented simplicity and flexibility. Features include automatic update of all indexes, multi-user support, an extensive range of validation, editing and masking facilities. Security at various levels is offered as an option. Accidents can be fatal — the UNIX and XENIX implementations of ZIM 2.5 will support audit trails in multi-user mode. A new utility is provided to use audit trails to recover the database in the event of system failure — insurance for your hours of work.



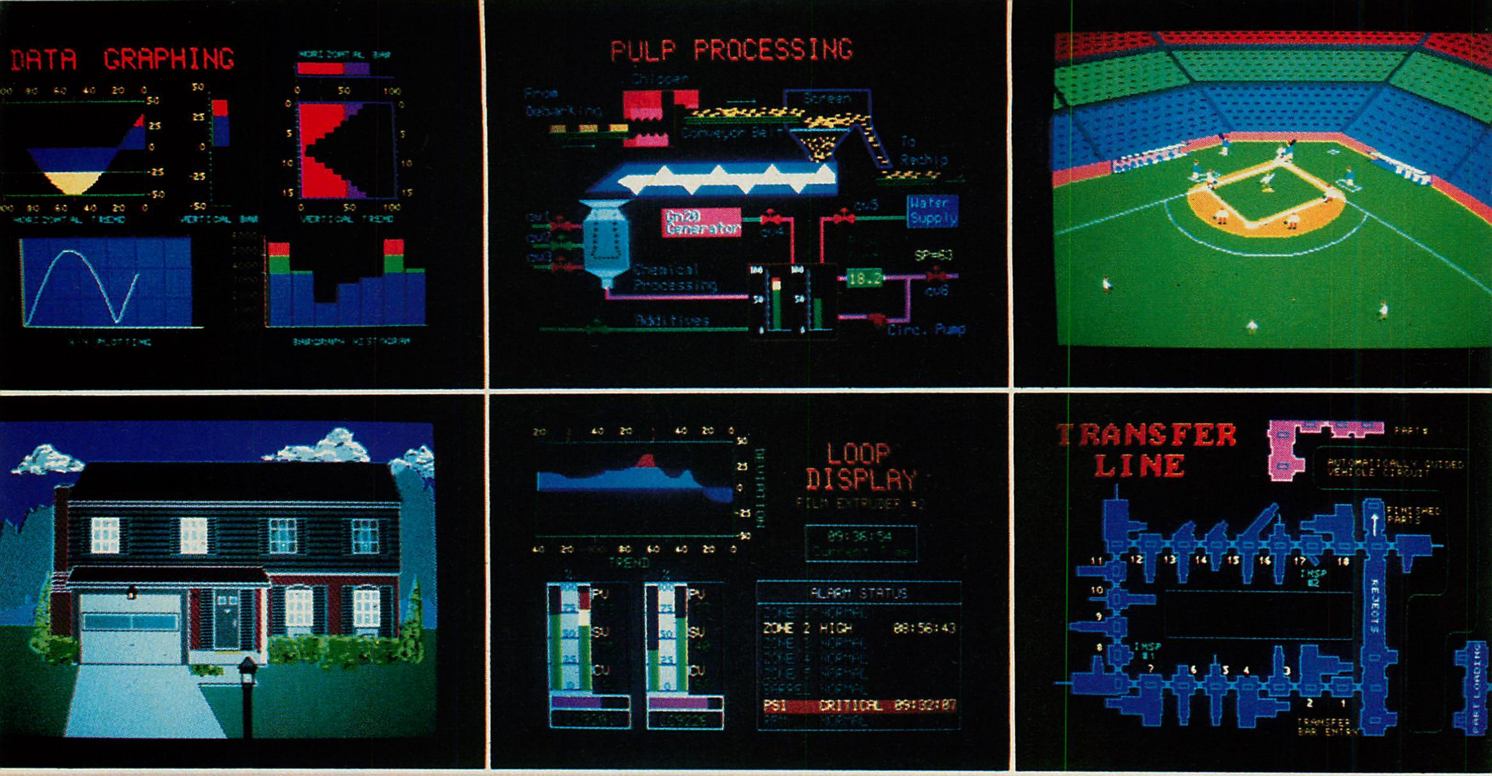
No matter what your demands, ZIM 2.5 will give you optimum power at a minimal price.

The one and only ZIM 2.5. A DBMS revolution.

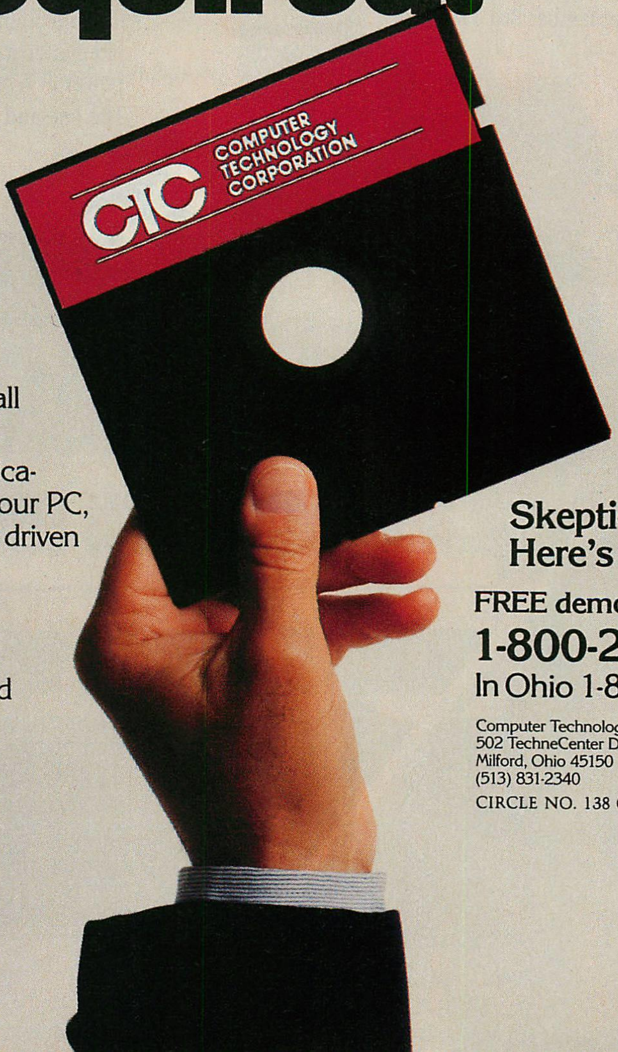
1785 Woodward Dr.
Ottawa Ontario
K2C 0R1
(613) 727-1397

ZANTHE
INFORMATION INC.

CIRCLE 148 ON READER SERVICE CARD



No Programming Required.



Create animated color graphics displays like these on your IBM PC in a matter of minutes...without any custom programming!

ScreenWare™ is an exciting new screen creation utility that's entirely software free, using a mouse to automatically generate all required code.

ScreenWare can be animated by any applications software co-resident in memory on your PC, or used as a stand-alone monitoring system driven by an external host computer.

OEM Agreements Available

Boost profits by customizing *your* standard applications software with ScreenWare!

Skeptical? Here's Proof!

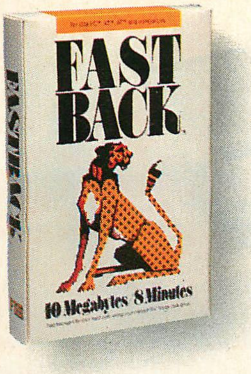
FREE demo disk. Call now

1-800-233-3329

In Ohio 1-800-255-3329

Computer Technology Corporation
502 TechneCenter Drive
Milford, Ohio 45150
(513) 831-2340

CIRCLE NO. 138 ON READER SERVICE CARD



Sleek Backup

An elegant solution to the problem of fixed-disk backup, Fastback is blemished only by its copy-protected status.

With programs that claim to do *everything* capturing a lot of attention, it is refreshing to note a program that does one simple task well. Simple does not necessarily mean easy. Consider, for example, the problem of fixed-disk backup. In the past, when the (now) lowly 10MB PC/XT-style fixed disk cost between $1,000 and $3,000, it was thinkable to spend $1,000 on a tape cartridge back-up system. Today, with that 10MB disk costing $450, the cost of tape backup is unreasonable.

The alternative of backing up fixed disks to floppy disks always has been available, but distasteful because the utilities for back up provided with PC-DOS are awkward and slow. A longtime empty niche—that spot waiting for a fixed-disk back-up utility with speed, an excellent user interface, and an attractive price—has now been filled. The Fastback fixed-disk back-up system by Fifth Generation Systems fits the bill and has been elected the *PC Tech Journal* Product of the Month for October.

Fastback is fast. Running in optimal mode on an XT with two floppy-disk drives in addition to the 10MB fixed disk, Fastback backed up 7.5MB of files in 6 minutes flat. A PC/AT with a single 1.2MB high-density floppy-disk drive backed up 9.4MB from its internal fixed disk to seven high-density floppy disks in 4 minutes 3 seconds. These figures are quite accurate because they were supplied by Fastback itself, which keeps track of time spent on the back-up operation. The PC/AT backup could have gone faster if two high-density floppy disks had been installed.

Fastback eschews a brute-force method without loss of speed. It does not make an image of the 10MB of fixed-disk storage, empty space and all; it copies file by file, subdirectory by subdirectory, from the fixed disk into a proprietary format on floppy disks. Fastback's organization enables it to fit more data (10 to 15 percent more) onto each disk than the standard 360KB or 1.2MB IBM scheme allows. In addition, Fastback formats blank disks on the fly while performing backups to them for the first time. The user can specify an individual subdirectory for back up, a file description using wildcards, or a backup of only those files that have changed since the last backup.

The first time Fastback is run on a system, it gathers information about the individual configuration through a question-and-answer session and writes it to a file. This saves it having to spend time sensing what sort of floppy-disk drive is being used (as with the PC/AT) or how many drives are present each time it is asked to perform a data backup. Additional time is saved by Fastback using true multitasking in its operation. It reads files from the fixed disk as fast as they can be read, and caches them into RAM buffers if it gets ahead of the process writing the data to floppy disk.

When a back-up operation is begun, a control panel appears on the screen in which every detail of the process is continually updated. A window is drawn for the system's fixed disk, plus two floppy-disk drives, in which instructions are placed for moving floppy disks into and out of disk drives.

PRODUCT NAME
Fastback

COMPANY
Fifth Generation Systems

ADDRESS
7942 Picardy Avenue, Suite B-350
Baton Rouge, LA 70809

TELEPHONE
504/767-0075

PRICE
$149

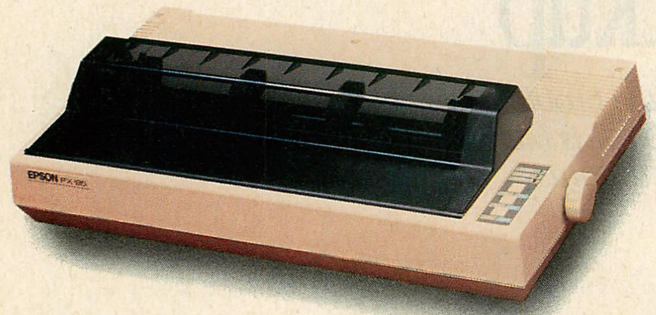
Operations can get lively on a system with two floppy-disk drives. Fastback prompts for a floppy in the second disk drive while writing data to the floppy in the first drive; when the disk in the first drive is full, it begins writing to the second without missing a beat. Thus, Fastback wastes no time waiting for human intervention, assuming the human can remove one floppy and insert another in the 17 seconds it takes to fill an entire 360KB disk. It is difficult to imagine how a disk back-up program might perform more efficiently than this.

When the backup is complete, a statistical summary of the operation is displayed, including the time required for the backup, the number of floppy disks required, the number of files backed up and the amount of disk space they occupy, the average number of bytes on each disk, and the average number of files on each disk.

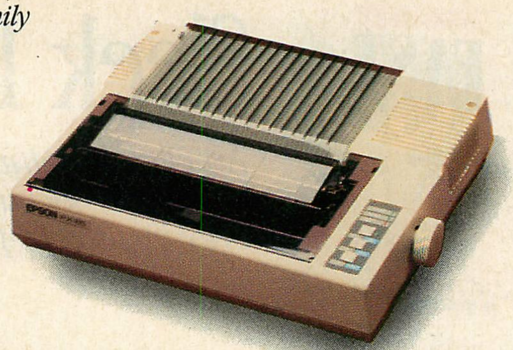
Fastback version 4.5 performed as discussed here. A subsequent release, 5.0, offers error correction improvements. The new version uses logic to correct one hard error per floppy disk track without affecting performance. Error-correcting abilities result in a more reliable emergency restore—far from bringing the restore to a halt, an occasional hard error will not even make the new Fastback break stride.

The only criticism to be made of Fastback, and it is substantial, is that it is copy-protected. While software piracy is a definite problem, is the ill will engendered by copy protection worth its unproven effectiveness? Should backup of irreplaceable data be entrusted to a program dependent on one fragile floppy disk? Each user will have to make his own decisions on this point. Prospective users of Fastback are encouraged to express their doubts to Fifth Generation Systems. Copy protection aside, however, Fastback is a most elegant solution to the knotty problem of fixed-disk backup for the PC.

Hardware, software, and other developments for the IBM PC family



Epson FX-185



Epson FX-85

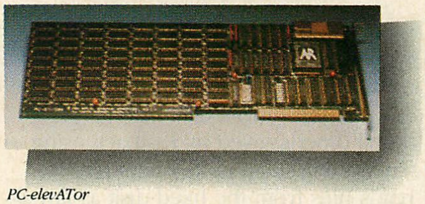
HARDWARE

The **Optical Disk Drive 5984** from **Optotech, Inc.** is a 5¼-inch, write-once-read-mostly (WORM) storage system with double-sided, 400MB removable cartridges that offer more than 200MB of on-line storage. The drive produces records that cannot be altered but can be updated using a proprietary "pointer field" that efficiently links previously stored information with new data. The 5984 provides approximately five times the capacity of equivalently priced Winchester hard disks and three times the capacity of magnetic tape. $5,000 for evaluation kit that includes drive, media controller, interface software, and documentation.
Optotech, Inc., 770 Wooten Road, Suite 109, Colorado Springs, CO 80915; 303/570-7500
CIRCLE 340 ON READER SERVICE CARD

A hardware/software package designed to capture and perform in-depth analysis of speech signals has been introduced by **Software Research Corporation**. Called **Micro Speech Lab**, the package allows researchers and instructors to perform signal input, waveform display, audio output, analysis, and file management. Users can "splice" a sound sample in both visual and auditory mode and condition the signal segment to required specifications. $1,350 ($1,800 Canadian).
Software Research Corporation, 3939 Quadra Street, Victoria BC, V8X 1J5, Canada; 604/727-3744
CIRCLE 341 ON READER SERVICE CARD

DYAD Technology Corporation has upgraded its **PC/VRTX** realtime multitasking operating system to support the PC/AT. PC/VRTX is a modular system component based on the versatile realtime executive (VRTX) made by Hunter and Ready, Inc. Plugged into a PC expansion slot, PC/VRTX provides 43 high-level system calls designed to manage the PC's I/O and memory resources and concurrently process as many as 256 applications programs tasks in realtime without programmer intervention. Two versions are available: PCX-2000, $1,495; PCX-1000 (with an additional 384KB of parity-protected memory), $1,695.
DYAD Technology Corporation, 4040-G Sorrento Valley Blvd., San Diego, CA 92121; 619/450-1761
CIRCLE 308 ON READER SERVICE CARD

PC-elevATor, from **Applied Reasoning Corporation**, combines Intel's 80286 chip on a plug-in board with software that enables a PC or PC/XT to emulate a PC/AT and return to PC or PC/XT mode at the push of a button. PC-elevATor runs software under DOS 2.x and 3.x five times faster, and programs for the PC/AT under DOS 3.x up to 50 percent faster than the AT itself. PC-elevATor can access up to 2MB of RAM. Configured with 512KB, $1,695; 1MB, $1,995; 2MB, $2,495.
Applied Reasoning Corporation, 84 Sherman Street, Cambridge, MA 02140; 617/492-0700
CIRCLE 310 ON READER SERVICE CARD

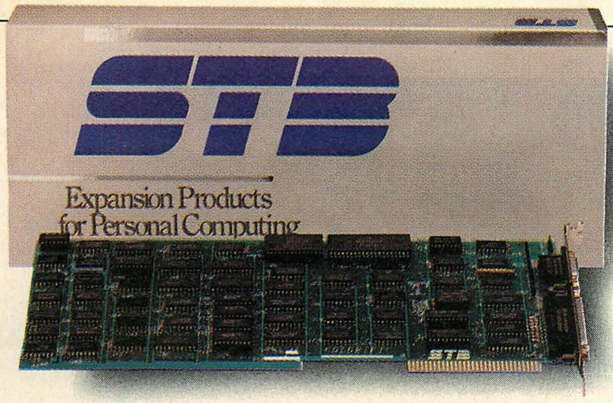


PC-elevATor

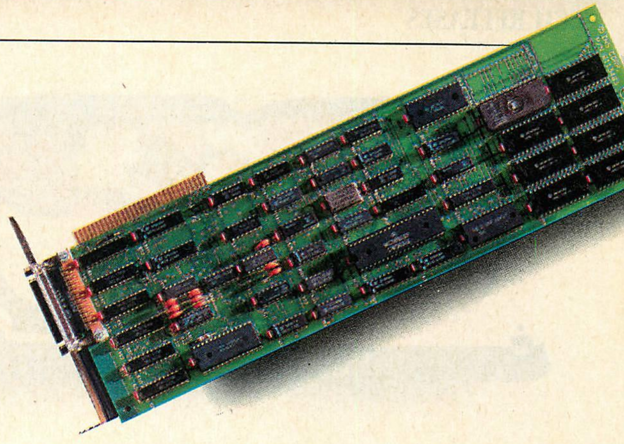
The next generation of FX printers has been introduced by **Epson America, Inc.** The 80-column **FX-85** and the 136-column **FX-185** offer high-speed draft and high-resolution, near-letter-quality copy, single-stroke selectable printing modes, and built-in IBM character sets. They come with an 8KB print buffer to free up computer time while the FX is printing. Both models feature 9-by-9 dot-matrix characters, logic-seeking bidirectional printing, 96 ASCII and 11 international character sets, and proportional space printing. In NLQ mode, users have a choice of pica, super/subscript, and expanded pica character sets. In draft mode, they may select pica or elite, or download their own character formats. FX-85, $499; FX-185, $699.
Epson America, Inc., Computer Products Division, 2780 Lomita Blvd., Torrance, CA 90505; 800/421-5426; in California, 213/539-9140
CIRCLE 309 ON READER SERVICE CARD

EXPANDER PLUS, from **Percom Data Corporation** is a modular, upgradable storage/expansion system that includes power control functions. The user may add hard disks, tape backups, removable hard disks, or more cards. The unit includes a 130-watt power supply, four AC receptacles and five power switches, AC surge suppressor, four card expansion slots, space for two 5¼-inch drives, and bus extension board/cables. $998; with Percom SUPERIOR INTERIOR 20MB drive (drive and controller card), $1,996.
Percom Data Corporation, 2703 National Place, Garland, TX 75041; 214/840-3032
CIRCLE 316 ON READER SERVICE CARD

Multi-AT, from **ANEX Technology, Inc.**, is an expansion system for the PC/AT. It permits eight workstations to access a single microprocessor by using a unique hardware/software combination that lets DOS recognize multiple workstations operating concurrently and transparent to each other; all users share a single set of software and common data and peripheral devices. The **Multi-PC**, a companion system to the Multi-AT, converts a PC/XT to a four-user system running under DOS. Multi-AT, $1,995; Multi-PC, $1,695.
ANEX Technology, Inc., 151 N. Route 9 W, Congers, NY 10920; 914/268-2400
CIRCLE 318 ON READER SERVICE CARD

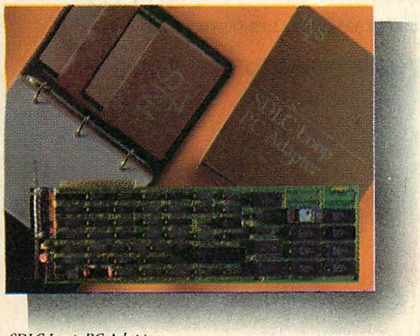


Colorific by STB



SDLC PC Adapter

Integrated Network Systems, Inc. offers two new products to better utilize the IBM SNA/SDLC network. The **SDLC PC Adapter** lets the PC attach via dial-up or leased line to any mainframe computer supporting SNA/SDLC, such as IBM's 4300 or S/370. The **SDLC Loop PC Adapter** provides attachment for the PC to the 8100 or 4331 SDLC communications loop. SDLC PC Adapter, $995; SDLC Loop PC Adapter, $1,295.
Integrated Network Systems, Inc., P.O. Box 91395, Mobile, AL 36691; 205/633-3270
CIRCLE 314 ON READER SERVICE CARD



SDLC Loop PC Adapter

Amdek Corporation has announced an ultra-high-resolution, dual-mode RGB video input color monitor, the **Color 722**, with an etched glass, nonglare .31 mm CRT and support for the dual-frequency output of the IBM Enhanced Graphics Adapter. In the 15.75-kHz mode, Color 722 emulates the IBM 5153 Color Display and compatible monitors, such as the the Amdek 600, to provide graphics resolution of up to 640 by 200 and as many as 16 colors. In the 21.8-kHz mode, it emulates the IBM Enhanced Color Display, providing graphics resolution of up to 620 by 350 and support for 16 colors from a palette of 64. $799; optional tilt/swivel stand, $29.
Amdek Corporation, 2201 Lively Blvd., Elk Grove Village, IL 60007; 312/364/1180
CIRCLE 317 ON READER SERVICE CARD

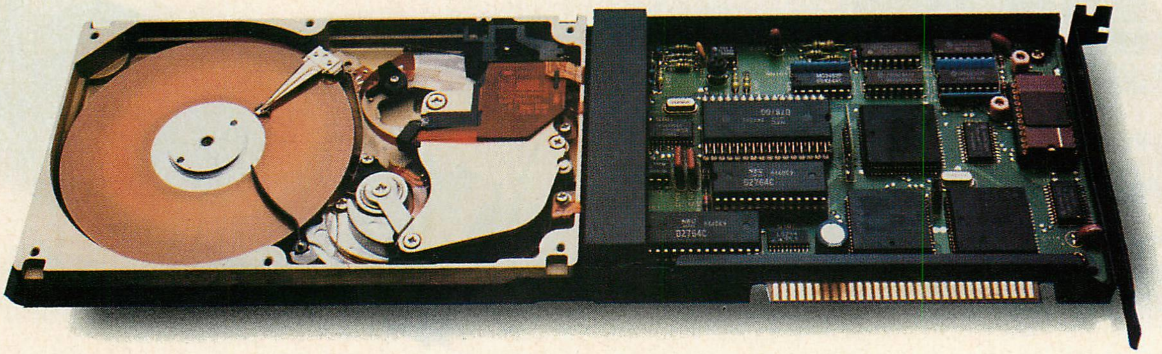
STB Systems, Inc. introduces **Colorific**, a new video board that emulates the IBM color/graphics display adapter, yet provides a parallel port, extended graphics capabilities, and an optional clock/calendar. With appropriate software, Colorific supports high-resolution display modes of 640-by-200-by-4 colors and 320-by-200-by-16 colors. The board offers 32KB of memory for extended graphics display; it is bundled with STB's PC Accelerator software, which provides drivers for the extended graphics display of Lotus 1-2-3, Symphony, and Framework. $275.
STB Systems, Inc., 601 N. Glenville, Suite 125, Richardson, TX 75081; 214/234-8750
CIRCLE 307 ON READER SERVICE CARD

A microcomputer-based, two-dimensional drafting system, **VISION 2000 Designer System**, has been announced by **Zericon**. The system comprises a graphics touch tablet and full-featured CAD software package with automatic dimensioning capabilities (a CAD pen plotter is optional). The VISION 2000 graphics touch table is used to select screen commands and position the drawing cursor; graphics can be created to 14 decimal places of accuracy. The pen plotter can work with paper sizes up to 24 inches by 36 inches with perfect curves and straight lines. CAD system, $495; robotic pen plotter, $395.
Zericon, 655 John Muir Drive, Suite 416, San Francisco, CA 94132; 415/585-9329
CIRCLE 311 ON READER SERVICE CARD

CXI has announced new accessories for 3270 micro-to-mainframe connections. The **Keyboard MATE** is a compact attachment that fits on the upper portion of the PC, PC/XT, or PC/AT keyboard and adds 3270 series and 3270-PC function keys; it enables CXI's PCOX customers to invoke 3270 functions with single keystrokes, eliminating most multiple-key functions for host and windowing operations. The **Daisy Chain Kit**, a modem bridge, permits serial attachment of multiple PCs to a single leased-line synchronous modem at a lower cost then equipping each PC with a modem. Keyboard MATE, $185; Daisy Chain Kit, $135; together, $250.
CXI, 3606 W. Bayshore Road, Palo Alto, CA 94303-4229; 415/424-0700
CIRCLE 315 ON READER SERVICE CARD

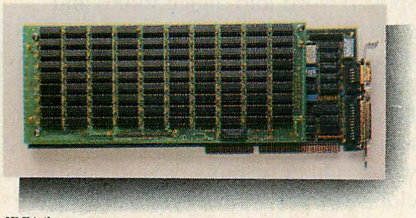
VoiceLink, from **Voice-Works, Inc.**, is a speaker-dependent voice recognition system that converts the sound of the spoken word into digital values and stores them on disk. During an application, the digital values are matched with the voice pattern of their originator. Equipped with a standard 128-word memory, VoiceLink is expandable to 512 words. Other features of VoiceLink include user-controlled program switching, a self-contained memory, 4800-baud asynchronous data transmission, a unidirectional headset microphone, and a user's guide. $1,100 ($1,700 Canadian).
Voice-Works, Inc., 74 Alex Avenue, Woodbridge, Ontario, L4L 4K6 Canada; 416/851-8543
CIRCLE 313 ON READER SERVICE CARD

BNW, Inc. has started shipping the latest versions of its Precision Graphics Adaptor high-resolution graphics board. **Model 12** can display 1,024-by-1,024 pixel resolution at a 50-kHz sweep rate. Its average line drawing rate is 500 nanoseconds per pixel with a maximum draw rate of 30 nanoseconds per pixel. **Model 15** will display 1,024 by 768 pixels at a 100-Hz, interlaced rate. The draw rate on the Model 15 is 700 nanoseconds per pixel. Models 12 and 15 have the same graphics primitives and are fully compatible. Model 12, $2,595; Model 15, $2,295.
BNW, Inc., 15951 Los Gatos Blvd., Suite 9, Los Gatos, CA 95030; 408/356-6148
CIRCLE 345 ON READER SERVICE CARD



Hardcard

A new high-performance multifunction board for the PC/AT basic or enhanced models has been introduced by **IDEAssociates, Inc.** The **IDEA Supermax** offers two serial ports and one parallel port and has a maximum memory capacity of 4MB. Included with Supermax is IDEA's menu-driven software. This software features complete diagnostics, print spooler, RAM disk, and desk calculator. Prices range from $495 for bare board to $3,795 with 4MB of memory.
IDEAssociates, Inc., 35 Dunham Road, Billerica, MA 01821; 617/663-6878
CIRCLE 319 ON READER SERVICE CARD



IDEA Supermax

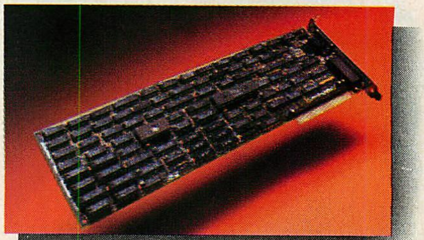
New Media Graphics Corporation has announced the **PC-VideoGraph**, a graphics add-on board for the PC, PC/XT, and PC/AT that allows high-quality videotaping of computer-generated graphics and text. PC-VideoGraph runs all standard IBM graphics and text generation software without reprogramming. The company provides optional PAINT and FONT software that allow easy creation of 640-by-400 resolution, 16-color graphics and text using a mouse or a light pen. PC-VideoGraph, $695; PAINT, $290; FONT, $190.
New Media Graphics Corporation, 279 Cambridge Street, Burlington, MA 01803; 617/272-8844
CIRCLE 322 ON READER SERVICE CARD

The **ADC-1** from **Remote Measurement Systems, Inc.** is a data acquisition and control system that is compatible with any computer equipped with an RS-232 interface. The system contains 16 13-bit analog-to-digital input channels, four digital inputs, six hardware controlled outputs, plus the ability to operate BSR-type, AC-line-carrier remote control modules. The high-resolution A/D conversion makes the ADC-1 suitable for laboratory and scientific data collection. $449.
Remote Measurement Systems, Inc., 2633 Eastlake Avenue E, Suite 206, Seattle, WA 98102; 206/328-2255
CIRCLE 320 ON READER SERVICE CARD

Plus Development Corporation has introduced **Hardcard**, the first hard disk on an IBM PC plug-in board. Hardcard comes with a 3½-inch hard-disk drive, electronics, and file management and installation software, all on an add-in card that is 4 inches by 13 inches by 1 inch. The two-pound board consumes 10.9 watts when operating and has an average access time of 65 milliseconds. Hardcard comes with standard 3370 heads, cobalt-doped oxide media, a rotary wedge servo actuator with an optical encoder, an air-lock system, and a SASI controller specially modified for the PC bus. $1,095.
Plus Development Corporation, 1778 McCarthy Blvd., Milpitas, CA 95035; 408/946-3700
CIRCLE 312 ON READER SERVICE CARD

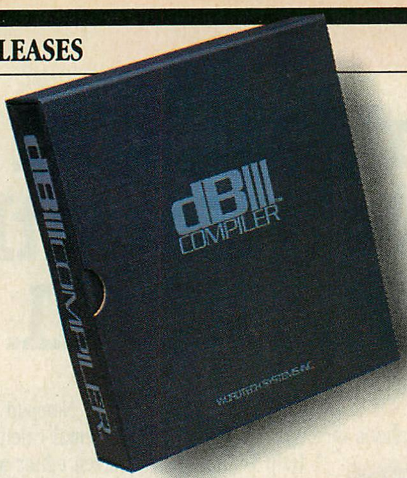
The newest member of **3Com Corporation**'s family of dedicated network servers, **3Server 70**, features a 70MB disk drive, providing almost twice as much disk storage as 3Server. The hardware integrates PCs and host systems into a LAN when used with 3Com's EtherSeries network products. Also announced is a 70MB **expansion disk**—an external disk drive that can be added to either server; both servers can handle 50 network users. 3Server 70, $8,995; expansion disk, $4,995.
3Com Corporation, 1365 Shorebird Way, P.O. Box 7390, Mountain View, CA 94039; 415/961-9602
CIRCLE 346 ON READER SERVICE CARD

AST Research, Inc. has introduced **ColorGraphPlus**, a display adapter for the PC that combines support for more colors than IBM's model 4190 Color/Graphics Monitor Adapter with a printer port. Occupying a single expansion slot, ColorGraphPlus supports two modes of operation: alphanumeric and all-points-addressable graphics. In its base configuration, the board provides as many as 16 colors in the medium-resolution, 320-by-200 mode. When operating in high-resolution, 640-by-200 mode, it supports four colors. $295.
AST Research, Inc., 2121 Alton Avenue, Irvine, CA 92714; 714/863-1333
CIRCLE 323 ON READER SERVICE CARD



ColorGraphPlus by AST

Racal-Vadic has introduced its **Multiple Data Set (MDS) II,** a system that allows users to manage modems in a corporate dial-up network from a local or remote site. With flexible configuration options, the MDS-II chassis will accommodate as many as 32 modems within a 19-inch rack, taking only 10½ vertical inches. The system controller performs complete central-site supervision and monitoring functions for a network of chassis, permitting worldwide control from a single location. VA1690 chassis with VA990 controller card and two VA 2190 power supplies, $3,260; VA4491E modem, $1,095; VA9000 system controller (interface board and software), $3,500.
Racal-Vadic, 1525 McCarthy Blvd., Milpitas, CA 95035; 408/946-2227
CIRCLE 321 ON READER SERVICE CARD



dBIIICompiler

dBASE III Developer's Release

SOFTWARE

IBM Corporation has announced its **BASIC compiler version 2.0,** which includes extensive enhancements while maintaining upward compatibility with 1.0. Version 2.0 runs under TopView; it supports the IBM PC Network environment, the Professional Color Display in compatibility mode, and the IBM Enhanced Graphics Adapter through language bindings and a device driver, available in the IBM PC Graphics Development Toolkit. The enhancements include named subroutines, user-defined multiline functions, the ability to compile larger programs, support for large numeric dynamic arrays, expanded graphics capabilities, expanded access to DOS, support for redirection of standard I/O, and support for lock/unlock file features. $495; update, $195. *IBM Corporation, Entry Systems Division, P. O. Box 1328, Boca Raton, FL 33432 (Contact the local IBM dealer)*

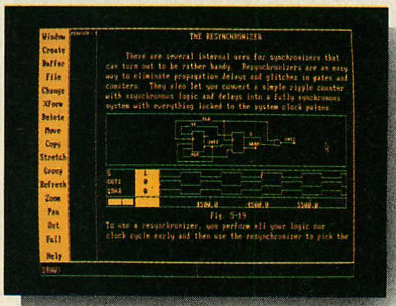
CIRCLE 324 ON READER SERVICE CARD

H.E.L.P. (*h*elp, *e*ditor, *l*int, *p*roductivity) is a multiwindow programming environment in C from **Everest Solutions, Inc.** Working from the editor, H.E.L.P. allows the user to pop up menus to perform tasks and editing, and to open as many windows as desired in which to edit or create C code, a letter, or notes. The user determines the size and position of the window; he can move among windows and move or copy text from one window to another. H.E.L.P. checks code for syntax problems and looks for inefficient use of memory, porting problems, and inconsistencies. When an error is found, the user is notified and given suggestions for correction; one keystroke corrects the error. $395. *Everest Solutions, Inc., 3350 Scott Blvd., Building 58, Santa Clara, CA 95051; 408/986-8977.*

CIRCLE 339 ON READER SERVICE CARD

Viewlogic Systems, Inc. has announced a new series of CAE packages called the **Workview Series**. They enable electronic design engineers to design, generate documentation, and communicate with mainframes, minicomputers, PCs, and other CAE/CAD workstations. Three versions of the Workview program make up the series: an entry-level system, a digital design system, and an analog design system. Features include schematic entry, interactive logic simulation and waveform processing, document processing that merges text and graphics, and communications facilities. Prices from $5,500 to $8,500. *Viewlogic Systems, Inc., 33 Boston Post Road W, Marlboro, MA 01752; 617/480-0881*

CIRCLE 342 ON READER SERVICE CARD



Workview screen

A version of **Ashton-Tates's dBASE III** designed specifically for applications developers is now available. **The Developer's Release** is packaged with a copy of the *Advanced Programmer's Guide*, including a disk containing the utilities, and Runtime+, a pair of utilities (dBCODE and dBLINKER) that allows developers to encrypt and link dBASE III custom applications. The Developer's Release includes eight new commands, 32 additional functions, and a new command-editing capability. $695. *Ashton-Tate, 10150 W. Jefferson Blvd., Culver City, CA 90230; 213/204-5570*

CIRCLE 330 ON READER SERVICE CARD

WordTech Systems, Inc. has introduced a new compiler for dBASE III called **dBIIICompiler**. It enables a programmer to generate encrypted applications programs; compiled programs run faster and require less storage space. Cross-environment linkers aid in producing code for different operating systems. dBIIICompiler supports the programming features of dBASE III. $750; cross-environment linkers: $100; upgrade from dBCompiler, $250. *WordTech Systems, Inc., P. O. Box 1747, Orinda, CA 94563; 415/254-0900*

CIRCLE 331 ON READER SERVICE CARD

VersaPro-3D has been announced by **MEGA CADD, Inc.** and **T&W Systems**. The product combines MEGA CADD's Design Board, a three-dimensional CAD package, and T & W Systems' VersaCad, a two-dimensional drafting package. VersaPro-3D lets users create complex three-dimensional models and generate perspectives, walkthrough sequences, isometrics, and other rotated views. These design drawings can be the basis for subsequent drafting tasks. $3,995. *MEGA CADD, Inc., Fifth Floor, The Court in the Square, 401 Second Avenue S, Seattle, WA 98104; 206/623-6245*

CIRCLE 343 ON READER SERVICE CARD

T&W Systems, Inc., 7372 Prince Drive, Huntington Beach, CA 92647; 714/847-9960

CIRCLE 347 ON READER SERVICE CARD

A Pascal screen-development utility for use with Turbo Pascal has been introduced by **PASCOM COMPUTING**. Called **TURBO SCREEN**, it permits rapid development of as many as 80 I/O screens to speed program development. TURBO SCREEN includes a screen editor, collator, and source generator, and supports most terminal attributes. $49.95. *PASCOM COMPUTING, 23611 Chagrin Blvd., Suite 101, Cleveland, OH 44122; 216/292-8745*

CIRCLE 334 ON READER SERVICE CARD

PC WEEK'S PRODUCT OF THE YEAR
PC MAGAZINE'S AWARD FOR TECHNICAL EXCELLENCE

Borland Introduces the Laws of *TURBO DYNAMICS*™

Laws That Work Like Magic. Whether considering technological excellence, or innovation in areas such as pricing, not copy-protection, licensing agreements, site licenses, 60 day money-back guarantee—Borland is clearly recognized as the software industry leader. The following three laws of "*Turbo Dynamics*"™ exemplify our pledge for excellence.

1ST LAW:

SPEED, POWER AND PRICE.

Borland products are known to be fast, powerful and to deliver an incredible price performance ratio. We only believe in absolutely superb software at rock bottom prices.

2ND LAW:

NOT COPY-PROTECTED SOFTWARE AND REASONABLE LICENSING AGREEMENTS.

We will always offer not copy-protected versions of our software. Also, our licensing agreement is now so simple that even a child can understand it.

3RD LAW:

60 DAY MONEY-BACK GUARANTEE.

This third law is actually a first in the industry! We are so sure that you will love our software that all of our products now come backed with a 60 day money-back guarantee. No questions asked.

Turbo Dynamics Applies to Turbo Pascal. Borland's Pascal family of products is growing by leaps and bounds. You can now join hundreds of thousands of users and enter the world of Turbo Pascal programming. And remember, all three laws of *Turbo Dynamics* apply to all Borland products.

TURBO PASCAL™ *$69.95*

The industry standard. With more than 350,000 users worldwide Turbo Pascal is the industry's de facto standard. Turbo Pascal is praised by more engineers, hobbyists, students and professional programmers than any other development environment in the history of microcomputing. And yet, Turbo Pascal is simple and fun to use. **Free spreadsheet** included on every Turbo disk with ready-to-compile source code. ***Options:*** We offer the exciting Binary Coded Decimal (BCD) option for your business applications as well as an 8087 option for your number-crunching applications at a very low charge. Please refer to the coupon. ***Portability.*** Turbo Pascal is available today for most computers running PC-DOS, MS-DOS, CP/M-80 or CP/M-86. ***Jeff Duntemann, PC Magazine:*** "In its simplicity it achieves an elegance that no other language compiler has ever displayed."

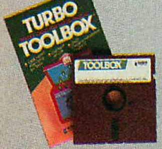
TURBO GRAPHIX TOOLBOX™ *$54.95*

High resolution monochrome graphics and window management for the IBM PC. The Turbo Graphix Toolbox will give even a beginning programmer the expert's edge. It's a complete library of Pascal procedures and functions. Tools that will allow you to draw and hatch pie charts, bar charts, circles, rectangles and a full range of geometric shapes. Procedures that will save and restore graphic images to and from disk. And much, much, more. You may incorporate part or all of these tools in your programs and yet we won't charge you any royalties. Best of all, these functions and procedures come complete with commented source code on disk ready to compile.

TURBO TUTOR™ *$34.95*

From start to finish in 300 pages. Turbo Tutor is for everyone from novice to expert. Even if you've never programmed before Turbo Tutor will get you started right away. ***A must.*** You'll find the source code for all the examples in the book on the accompanying disk ready to compile. Turbo Tutor might be the only reference on Pascal and programming you'll ever need.

TURBO DATABASE TOOLBOX™ *$54.95*



The Turbo Database Toolbox is the perfect complement to Turbo Pascal. It contains a complete library of Pascal procedures that allows you to sort and search your data and build powerful applications. It's another Borland set of tools that will give the beginning programmer the expert's edge. ***Get started right away: free database!*** Included on every Toolbox disk is the source code to a working data base which demonstrates how powerful and easy to use our search system, Turbo-Access, really is. Modify it to suit your individual needs or just compile it and run. ***Remember, no royalties!***

NOT COPY-PROTECTED

TURBO PASCAL FAMILY

Available at better dealers nationwide. Call (800) 556-2283 for the dealer nearest you. To order by Credit Card call (800) 255-8008, CA (800) 742-1133

60 DAY MONEY-BACK GUARANTEE

Carefully Describe your Computer System!

Mine is: ___ 8 bit ___ 16 bit
I Use: ___ PC-DOS ___ MS-DOS
___ CP/M 80 ___ CP/M 86

My computers' name/model is: ________

The disk size I use is:
☐ 3½" ☐ 5¼" ☐ 8"

Name: ________
Shipping Address: ________
____
City: ________
State: ________ Zip: ________
Telephone: ________

Pascal 3.0	$ 69.95	____
Pascal w/8087	$109.90	____
Pascal w/BCD	$109.90	____
Pascal w/8087 & BCD	$124.95	____
Turbo Database Toolbox	$ 54.95	____
Turbo Graphix	$ 54.95	____
Turbo Tutor	$ 34.95	____

*These prices include shipping to all U.S. cities. All foreign orders add $10 per product ordered.

Amount: (CA 6% tax) ________

Payment: VISA MC BankDraft Check

Credit Card Expir. Date: ____/____

Card #: ________

F6

COD's and Purchase Orders WILL NOT be accepted by Borland. California residents: add 6% sales tax. Outside USA: add $10 and make payment by bank draft, payable in US dollars drawn on a US bank.

BORLAND INTERNATIONAL

4585 Scotts Valley Drive, Scotts Valley CA 95066
Phone (408) 438-8400 Telex 172373

Software Development Tools

If:

Manufacturer-Compatible, Fully-Supported Cost-Effective, Cross and Native Development tools are important to you!

Then:

Let us tell you about MICROTEC RESEARCH's extensive line of field-proven software development tools for serious software developers.

C
Pascal
PLM
Assemblers
Simulators
VT-100 Emulator
CPM/80 Emulator

Manufacturer Compatible

MICROTEC RESEARCH has been providing flexible and economical solutions for software developers since 1974. We've grown with the industry as our cross software development tools have transformed many large and small computers into powerful cross development systems for microprocessor applications. Our software tools are manufacturer compatible. Therefore, software made using manufacturers development systems may be more productively used in the MICROTEC RESEARCH cross-development environment.

Fully-Supported

MICROTEC RESEARCH's products are continually maintained and updated based upon the experience of thousands of installations worldwide. Revisions and modifications are distributed to licensees in warranty or covered by a service contract. Upgrades, which are major enhancements to a product's capabilities, are available at a significant discount. Our update/upgrade policy assures users they'll always be current with the fast moving world of microprocessor development.

Effective Differences

Beginning with product concept, through development, quality assurance, and post-sales support — **Quality, Compatibility,** and **Service** are the differences which set MICROTEC RESEARCH apart from the others.

Start Saving Time & Money

If you are a serious software developer, shopping for software development tools, call or write today for more information. **800-551-5554** In CA call **(408) 733-2919**

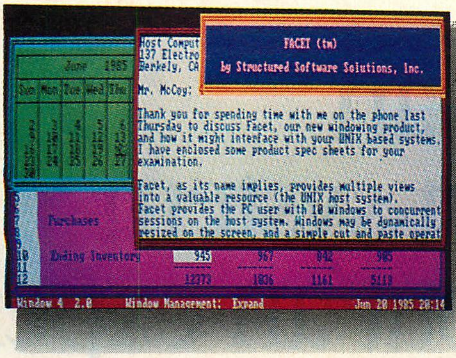
Target Microprocessors		Host Computers	Software Tools
8086/80186	68000/08/10	DEC VAX, PDP-11	C Compilers
8096	6809	DG MV-Series	Pascal Compilers
8080/8085	6800/01/02	DG Eclipse	PLM Compiler
8051	6805	Apollo	Assemblers
8048	6301	UNIX Systems	Linking Loaders
Z8002	6305	IBM PC	Librarians
Z80	6500	Data General/ONE	Download
Z8	G65SCXX, G65SC1XX	HP 150	Communications
NSC 800	R65C00, R6500/1	DEC Rainbow	VT-100 Emulator
9900...	others	others	CPM/80 Emulator

for Serious Software Developers

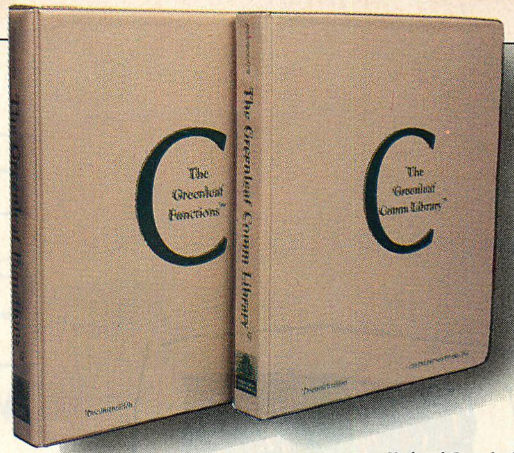
MICROTEC RESEARCH

3930 Freedom Circle, Suite 101, Santa Clara, CA 95054
Mailing Address: P.O. Box 60337, Sunnyvale, CA 94088
(408) 733-2919 • Telex (ITT) 4990808

CIRCLE NO. 214 ON READER SERVICE CARD



FACET screen



Updated Greenleaf C Libraries

Three new entries in the bulletin board software arena are **BBS-PC** from **Micro-Systems Software, Inc.**, CHANNEL ONE from **Modem Controls, Inc.**, and **The Bread Board System** from **eSoft,Inc.** BBS-PC supports electronic mail and program or data file exchanges. It provides 16 separate sections (subboards), four file transfer protocols, and a terminal-mode feature that supports a second modem, COM port, and telephone line to permit answering one modem and dialing out on the other. CHANNEL ONE offers electronic carbon copying, mailing lists, and the ability to accept files in worksheet (binary) format without the need for conversion. The Bread Board System offers a menu editor for custom system appearance and operation, word wrap, many protocols for file and data transfer, a capacity for 3,500 messages on as many as 25 separate message boards, and system security by the optional use of a log-on password check. BBS-PC, $249.00; CHANNEL ONE, $495.00; The Bread Board System, $299.95.
Micro-Systems Software, Inc., 4301-18 Oak Circle, Boca Raton, FL 33431; 305/391-5077
CIRCLE 325 ON READER SERVICE CARD
Modem Controls, Inc., 320 N. Michigan Avenue, Suite 2100, Chicago, IL 60601; 312/346-4600
CIRCLE 326 ON READER SERVICE CARD
eSoft, Inc., 4100 S. Parker Road, Suite 305, Aurora, CO 80014; 303/699-6565
CIRCLE 327 ON READER SERVICE CARD

Structured Software Solutions, Inc. has announced **FACET**, a distributed windowing environment that runs on the PC and compatibles and communicates with a companion driver on a UNIX or XENIX host computer. It features multiple modes of window selection, automatic window overlap, window borders, Caps and Num Lock settings, color monitor support, support of X.PC multisession protocol, cut-and-paste capabilities, and the ability to configure as many as 10 windows, each with interactive sessions. Windows may be expanded, contracted, panned, and moved on the screen. PC host computer with four users, $195; PC/AT host computer with four users, $249.
Structured Software Solutions, Inc., 4031 W. Plano Parkway, Suite 205, Plano, TX 75075; 214/985-9901
CIRCLE 329 ON READER SERVICE CARD

Rabbit Software Corporation has announced **SNA-PLUS**, an advanced implementation of IBM's SNA LU 6.2 protocol. SNA-PLUS is designed to connect non-IBM products, including micros, minis, and computer network systems, with LU 6.2, the latest SNA enhancement for micro systems. It lets users communicate with any application, no matter what language it is written in, and access data on any processor, anywhere on the network. Prices vary depending on the number of computers in the network.
Rabbit Software Corporation, Great Valley Corporate Center, One Great Valley Parkway E, Malvern, PA 19355; 215/647-0440
CIRCLE 344 ON READER SERVICE CARD

Design Graphics Software (DGS) is an engineering design software system for the PC, PC/XT, and PC/AT from **The CADWARE Group, Ltd.** that combines expert system methodology and interactive computer graphics. DGS is based on an open system, applications independent architecture and employs rule-based methodology that deals with graphically specified design rules and usage procedures. DGS may be used in mechanical design, process control proposal writing, smart forms, and architectural design. It may be operated in either stand-alone or networked configurations. $3,500.
The CADWARE Group, Ltd., 869 Whalley Avenue, New Haven, CT 06515; 203/397-2908
CIRCLE 338 ON READER SERVICE CARD

Greenleaf Software, Inc. has introduced new versions of its function libraries for the C language. **Version 3.0** of **Greenleaf Functions** features speed and code density, as a result of many functions being coded in assembly language, and a dozen new functions, such as determining when a drive door is open. The **Greenleaf Comm Library** has interrupt-driven asynchronous communications for as many as eight ports simultaneously, up to 9600 baud, with Hayes modem controls, and support for XMODEM and XON-XOFF protocols. Both products support the Microsoft C 3.0 and Wizard Systems MSDOS C compiler 2.0, as well as Lattice C, Computer Innovations C86, DeSmet C, and Mark Williams MW86. All memory models are supported. $185 for each library.
Greenleaf Software, Inc., 1411 LeMay Drive, Suite 101, Carrollton, TX 75007; 214/446-8641
CIRCLE 332 ON READER SERVICE CARD

Microsoft Windows, by **Microsoft Corporation**, is an extension to MS-DOS that allows users to run multiple DOS-based applications programs at the same time. It facilitates data interchange among different programs and provides a graphics-based user interface to simplify interaction with the operating system. Also available to software developers are the **Microsoft Windows Developer Tool Kit** and a **runtime version** of the Windows product. This software provides information necessary to applications development that takes advantage of the Microsoft Windows data interchange facilities, memory management features, graphics capabilities, and user interface. Contact the company for information on the runtime version. Developer Tool Kit, $500.
Microsoft Corporation, 10700 Northup Way, Box 97200, Bellevue, WA 98009; 206/828-8080
CIRCLE 328 ON READER SERVICE CARD

Visual: GeniSys screen

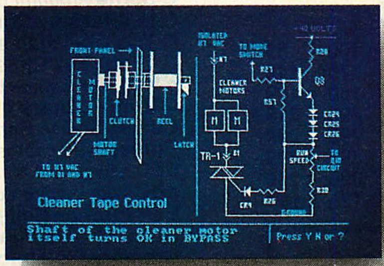
Knight Data Security Manager

Now available for the PC, PC/XT, and PC/AT is **Softool Corporation's Fortran Programming Environment (FPE)**, an integrated set of programmer productivity tools that includes structured languages, a source code documenter, interface documenter, error detectors, libraries of prefabricated code, testing aids, tracing aids, optimizers, communication software, as well as tutorials. FPE allows programmers to produce a product that is fully tested, documented, and standardized. $1,495.
Softool Corporation, 340 S. Kellogg Avenue, Goleta, CA 93117; 805/683-5777
CIRCLE 333 ON READER SERVICE CARD

Visual Engineering has announced **Visual:GeniSys**, a high-speed three-dimensional program for solids modeling that features constant-time ray-tracing technology. It includes anti-aliasing, a hierarchical database, and simulated camera and lighting control. The integrated rendering system allows complex three-dimensional objects and entire scenes to be generated from data, giving the user control over trade-offs between realism and compute time. Visual:GeniSys is offered with a subroutine library interface and an optional textual file interpreter. $10,000 to $75,000 depending on system configuration.
Visual Engineering, 2680 N. First Street, Suite 200, San Jose, CA 95134; 408/945-9055
CIRCLE 336 ON READER SERVICE CARD

From **AST Research, Inc.** comes a software package that offers complete data security management, convenient DOS interface, usage auditing, and electronic mail capabilities for a single PC in individual or shared environments. **Knight Data Security Manager** supports all DOS-compatible applications software. For the PC/XT and PC/AT, $295.
AST Research, Inc., 2121 Alton Avenue, Irvine, CA 92714; 714/863-1333
CIRCLE 337 ON READER SERVICE CARD

KDS (Knowledge Delivery System), by **KDS Corporation**, is an artificial intelligence software program that produces expert systems. These expert systems permit the nonexpert end user to use the knowledge of an expert for diagnosis, decision support, and administrative management. KDS produces as many as 16,000 rules from 4,096 case histories per knowledge module. As many as 512 conditions may be used in any combination to distinguish one case history from another. Knowledge modules may be linked to produce systems that are limited only by the disk capacity. $795; playback utility, $150 with KDS.
KDS Corporation, 934 Hunter Road, Wilmette, IL 60091; 312/251-2621
CIRCLE 335 ON READER SERVICE CARD



Sample KDS screen

The **ES/P Advisor**, now written in Prolog 2, permits users with no prior knowledge of artificial intelligence programming languages or techniques to design and implement useful, practical expert systems. Produced by **Expert Systems International**, ES/P benefits from the advanced features of Prolog 2, such as general purpose interfacing, faster program execution, virtual memory management, and windows for character-level editing, in delivering performance that is four to five times faster than the original version. $895.
Expert Systems International, 1150 First Avenue, King of Prussia, PA 19406; 215/337-2300
CIRCLE 350 ON READER SERVICE CARD

X-VIEW 86 is a technical tool from **McGraw-Hill, Inc.** that allows hardware and software to observe the internal operations of DOS applications software. It is a software analyzer consisting of a diskette, documentation, and technical support for registered users. The technical specialist can interrupt an applications program on any processor I/O access or interrupt instruction, or when its execution reaches a specified address or it uses a specified memory location. X-VIEW 86 can automatically collect and analyze previously unavailable technical data, start program execution from a specified address, and display the results of the analysis on the screen. It can be used to improve program performance, facilitate conversion and porting of software, identify and eliminate unwanted side effects, discover code hotspots, and find program bugs. Accompanying documentation includes numerous examples and suggested uses; seven technical appendices provide such information as I/O port assignments for the PC, PC/XT, PC/AT, and PC*jr*, and the PC family BIOS interrupt and function calls. $59.95.
McGraw-Hill, Inc., 8111 LBJ Freeway, Dallas, TX 75251; 800/221-VIEW; in Texas, 800/233-VIEW
CIRCLE 349 ON READER SERVICE CARD

Gimpel Software has announced the availability of **PC-Lint**, a diagnostic facility for the C language running under MS-DOS. PC-Lint reports type inconsistencies across modules, parameter argument mismatches, library usage irregularities, uninitialized variables, value-return inconsistencies, variables declared but not used, suspicious use of operators, and unreachable code. PC-Lint is delivered with user-modifiable standard library descriptions for most major compilers. $139.
Gimpel Software, 3207 Hogarth Lane, Collegeville, PA 19426; 215/584-4261
CIRCLE 348 ON READER SERVICE CARD

PROGRAMMER DEVELOPMENT TOOLS

We now accept
Corporate Accounts & P.O.'s
from qualified companies and educational institutions
at NO EXTRA CHARGE.

Here are some additional reasons to buy from Programmer's Connection.

1. We offer the latest versions of products.
2. Most popular products are in stock ready for shipment.
3. Receive same manufacturer's support as if buying direct.
4. Experienced professional programmers are on staff.
5. Select from a wide variety of the best software products available.
6. Knowledgeable and courteous sales staff.
7. Significant discounts off retail prices.
8. No extra charge for use of major credit cards.
9. Reasonable charges for shipping and handling.
10. Toll free service from Canada and the United States.

APL AND UTILITIES

	LIST	OURS
APL*PLUS/PC System by STSC	595	449
APL*PLUS/PC Tools Vol 1	295	239
APL*PLUS/PC Tools Vol 2 ... *New*	Call	Call
APL2C by Lauer Software ... *Interfaces APL to C*	150	139
Financial/Statistical Library by STSC	275	219
Pocket APL by STSC ... *New & Better*	95	89
STATGRAPHICS by STSC	695	539

ARTIFICIAL INTELLIGENCE

	LIST	OURS
Golden Common LISP by Gold Hill	495	Call
Prolog-86 by Solution Systems	125	Call

ASSEMBLERS AND DEBUGGERS

	LIST	OURS
8088 Assembler *w/Z-80 Translator* by 2500 AD	100	89
Advanced Trace-86 by Morgan Computing	175	149
Microsoft Assembler *w/utilities* by Microsoft	150	109
Pasm86 *High Perf Assembler* by Phoenix	295	219
Periscope *Debugger* by Data Base Decisions	295	269
Pfinish *Performance Analyzer* by Phoenix	395	299
Pfix-86 Plus *Symbolic Debugger* by Phoenix	395	299
Profiler by DWB Associates ... *New Version*	125	109

CodeSmith™-86 by Visual Age
This debugger with dynamic patching assembler now includes File Browser/Editor windows with synchronized Source Code Display for Microsoft C and Pascal.
List Price $145 Our Price **$109**
An optional hardware breakout switch is also available.

BASIC LANGUAGE

	LIST	OURS
BetterBASIC by Summit Sotfware	200	169
8087 Math Support	99	89
Btrieve Interface ... *Used with Btrieve*	99	89
Run-time Module	250	239
Microsoft QuickBASIC Compiler ... *New*	99	79
Professional BASIC by Morgan Computing	99	89
8087 Math Support	50	47
True Basic from Addison-Wesley	150	129
Run-time Module	500	459

BLAISE PRODUCTS

	LIST	OURS
Asynch Manager *for C or Pascal*	175	139
C Tools	125	109
C Tools 2	100	89
Exec Linker *Program Chainer*	95	84
Pascal Tools	125	109
Pascal Tools 2	100	89
Turbo ASYNCH *for Turbo Pascal*	100	89
Turbo POWER TOOLS *for Turbo Pascal*	100	89
View Manager *for C or Pascal*	275	219
View Manager *with Source*	425	359

C COMPILERS

	LIST	OURS
C-86 Compiler by Computer Innovations	395	289
DeSmet C Compiler *w/Source Debugger*	159	145
Lattice C Compiler from Lattice	500	349
Lattice C from Lifeboat	500	299
Let's C *w/Source Debugger* by Mark Williams	150	129
Microsoft C Compiler *version 3* ... *Sale*	395	259
Wizard C Compiler by Wizard Systems *Includes lint*	450	399

Mark Williams MWC-86 Version 3.0 with Source Level Debugger
Mark Williams Company
The new version of this super C compiler now comes with UNIX-like utilities including make, diff, m4, ED, MicroEMACS, egrep and more.
List price $495 Sale price **$379**

C INTERPRETERS

	LIST	OURS
C-terp *C Interpreter* by Gimpel Software	300	249
Instant C by Rational Systems ... *Sale*	500	399
Introducing C by Computer Innovations	95	89
Run/C from Lifeboat	150	109

C UTILITIES

Call us for availability of specific compiler interfaces.

	LIST	OURS
Basic_C Library by C Source	175	139
Btrieve by Softcraft	250	199
C-Food Smorgasbord by Lattice	150	119
C-Sprite *Program Debugger* by Lattice	175	149
C to dBase by Computer Innovations	150	139
C Tools by Blaise Computing	125	109
C Tools 2 by Blaise Computing	100	89
c-tree *with source* by FairCom ... *New version*	395	339
C Utility Library by Essential Software	185	139
Curses *Screen Manager* by Lattice	125	109
Curses *with source code*	250	209
dBC *dBase File Manager for C* by Lattice	250	209
dBC *with source code*	500	439
ESP for C by Bellesoft ... *Sale*	349	229
GraphiC by Scientific Endeavors	250	209
Greenleaf C Functions Library	185	139
Greenleaf Comm Library	185	139
The HAMMER by OES Systems	195	179
MetaWINDOWS by Metagraphics	150	139
Multi-Halo *Graphics* by Media Cybernetics	250	199
NIMBUS *Business Graphics* by Media Cybernetics	195	179
PANEL by Roundhill ... *Library Source Available*	295	234
PC Lint by Gimpel Software	100	89
Pre-C *Lint Utility* by Phoenix	395	299
Scientific Subroutine Library *for C* by Peerless	175	139
Windows for C by Vermont Creative Software	195	139
Windows for Data by Vermont Creative Software	295	259

APT Application Programmer's Toolkit
This C library features direct and keyed file access, full-screen data input and editing, screen and report generation, string math, terminal drivers, file editing and many other useful utilities. Includes source code and no royalties.
List Price $395 Our Price **$339**

Programmer's
136 Sunnyside Street Hartville, Ohio 44632
Call For Our Catalog
CIRCLE NO. 175 ON READER SERVICE CARD
We accept qualified Corporate Accounts and P.O.'s. Dealer

FOR THE IBM-PC/XT/AT and compatibles.

1085

CROSS ASSEMBLERS

We carry a full line of 2500 AD cross assemblers. Call us with your specific needs.

	LIST	OURS
6502 XASM by 2500 AD	200	165
8080 XASM by 2500 AD	200	165
68000 XASM by 2500 AD	300	249
80186 XASM by 2500 AD	200	165
Z-80 XASM by 2500 AD	200	165
Z-8000 XASM by 2500 AD	300	249

FORTRAN COMPILERS AND UTILITIES

Microsoft Fortran .... *Links with Microsoft C*	350	239
RM/Fortran by Ryan-McFarland	595	439
Btrieve by SoftCraft	250	199
FORLIB-PLUS by Alpha Computer Service	70	59
Multi-Halo *Graphics* by Media Cybernetics	250	199
PANEL *Screen Designer* by Roundhill	295	234
PolyFortran Tools by Polytron	179	159
Scientific Subroutine Library by Peerless	175	139
The Statistician by Alpha Computer Service	295	269
Strings and Things by Alpha Computer Service	70	59

LATTICE PRODUCTS

All products in this section have Lattice serial numbers. These products receive support and updates directly from Lattice Inc., the company that actually developed the Lattice C compiler.

Lattice C is in stock and ready for shipment.

Lattice C Compiler	500	349
C-Food Smorgasbord *Function Library*	150	119
C-Sprite *Debugger* ...... *Now Supports Source Level*	175	149
Curses *Screen Manager*	125	109
Curses *with source code*	250	219
dBC *dBase File Manager for C*	250	219
dBC *with source code*	500	429
LMK *Make Facility*	195	159
Text Mgmt Utilities *(GREP/DIFF/ED/WC/Extract/Build)*	120	105

MICROSOFT C AND UTILITIES

Microsoft C Compiler *version 3* .... *Sale*	395	259
Blaise C Tools	125	109
Blaise C Tools 2	100	89
C-terp by Gimpel Software	300	249
c-tree *with source* by Faircom .... *New version*	395	339
C Utility Library by Essential Software	185	139
Greenleaf C Functions Library	185	139
Greenleaf Comm Library	185	139
The HAMMER by OES Systems	195	179
Microsoft Windows .... *New*	Call	Call
Multi-Halo *Graphics* by Media Cybernetics	250	199
PANEL by Roundhill ...... *Library Source Available*	295	234
Pre-C *Lint Utility* by Phoenix	395	299
Windows for C by Vermont Creative Software	195	139

Lattice dBC and C-Sprite now also support Microsoft C.

OTHER LANGUAGES

Janus/ADA C Pack by R&R Software	95	89
Janus/ADA D Pack by R&R Software	900	699
Level II COBOL by Micro Focus	Call	Call
Microsoft Pascal ..... *Links with Microsoft C*	300	219
Modula-2/86 by Logitech	495	439
PC/Forth+ by Laboratory Microsystems	250	209

PHOENIX PRODUCTS

Phoenix™ **authorized dealer**

In stock and ready for immediate shipment.

Pasm86 *High Performance Macro Assembler*	295	219
Pfinish *Performance Analyzer*	395	299
Pfix-86 Plus *Symbolic Debugger for Plink-86*	395	299
Plink-86 *Overlay Linker*	395	299
Pmaker *Program Development Manager*	195	139
Pmate *Macro Text Editor* .... *New version*	225	159
Pre-C *Lint Utility*	395	299

POLYTRON PRODUCTS

	LIST	OURS
Polytron C Library	99	89
PolyFortran Tools by Polytron	179	159
PolyLibrarian *Library Manager*	99	89
PolyLibrarian II *Library Manager*	149	129
PolyMake *UNIX-like Make Facility*	99	89
PolyOverlay *Overlay Optimizer*	99	89
PolyXREF *Cross Reference Utility* .... *New*	Call	Call
PVCS *Polytron Version Control System* ...... *New*	395	359
PVMFM *Polytron Virtual Memory File Mgr* ...... *New*	199	179

SOFTCRAFT PRODUCTS

Btrieve *ISAM File Manager*	250	199
Btrieve/N *for Networks*	595	469
Rtrieve *Report Generator for Btrieve*	85	79
Xtrieve *Query Utility for Btrieve*	195	169
Xtrieve/N *Query Utility for Btrieve/N*	395	299
New: **OPT-Tech Sort** now works with Btrieve	99	87

SOFTWARE HORIZONS PRODUCTS

C POWER PACKS:

Pack 1: **Building Blocks I**	149	129
Pack 2: **Database**	399	339
Pack 3: **Communications**	149	129
Pack 4: **Building Blocks II**	149	129
Pack 5: **Mathematics I**	99	87
Pack 6: **Utilities I**	99	87

TEXT EDITORS

Brief by Solution Systems	195	Call
Epsilon *Emacs-like Text Editor* by Lugaru	195	179
ESP for C by Bellesoft .... *Sale*	349	229
ESP for Pascal by Bellesoft .... *Sale*	249	169
ESP for C and Pascal by Bellesoft .... *Sale*	399	279
FirsTime for C by Spruce Technology	295	239
FirsTime for MS Pascal by Spruce Technology	245	199
Pmate *Macro Text Editor* by Phoenix ... *New version*	225	159
PS Tech Word Processor by Morgan Computing	695	599
Vedit by Compuview	Call	Call
XTC *Text Editor* by Wendin .... *Includes source*	99	89

TURBO PASCAL AND UTILITIES

Turbo PASCAL by Borland international	70	55
Turbo PASCAL *w/8087 or BCD*	110	95
Turbo PASCAL *w/8087 & BCD*	125	105
FirsTime for Turbo By Spruce Technology	75	69
Screen Sculptor by Software Bottling	125	99
Turbo ASYNCH by Blaise Computing	100	89
Turbo GRAPHIX TOOLBOX by Borland Int'l	55	49
Turbo POWER TOOLS by Blaise Computing	100	89
Turbo TOOLBOX by Borland Int'l	55	49
Turbo TUTOR by Borland Int'l	35	29
TurboPower Utilities *w/source* by TurboPower	95	89
TurboWindow by MetaGraphics	55	49
XTC *Text Editor* by Wendin .... *Includes source*	99	89

XENIX SYSTEM V BY SCO

Xenix 86 Development System .... *For XT*	595	549
Xenix 86 Operating System .... *For XT*	595	549
Xenix 86 Text Processing Package ...... *For XT*	495	439
Complete Xenix 86 System .... *Combined Package*	1350	1099
Xenix 286 Development System .... *For AT*	Call	Call
Xenix 286 Operating System .... *For AT*	Call	Call
Xenix 286 Text Processing Package ...... *For AT*	Call	Call
Complete Xenix 286 System ... *Combined Package*	Call	Call

XENIX LANGUAGES AND UTILITIES

APL*PLUS/UNX System by STSC ..... *For AT only*	995	795
c-tree by FairCom	Call	Call
Microsoft BASIC Interpreter	350	279
Microsoft Fortran	495	389
Microsoft PASCAL	495	389
PANEL *Screen Designer* by Roundhill ... *For AT only*	595	539
Windows for C by Vermont Creative Software	395	359
Windows for Data by Vermont Creative Software	595	539

PC Connection

VISA MasterCard

U.S.: 1-800-336-1166

Account is charged when order is shipped. Prices are subject to change without notice. Please include credit card expiration date when ordering by mail.

inquires invited. CANADA: 1-800-225-1166 OHIO: 1-216-877-3781

ALLEN DeLOACH

Custom Characters

Creating graphics for the high 128 characters of the PC's extended set is easy using Turbo Pascal's absolute address functions.

The PC's extended character set contains 256 legal values, all of which are defined in a character-generator ROM that is used in text mode. This ROM does not exist in the PC's memory map; only the video board's 6845 chip can access it. However, when the PC is operating in graphics mode, the character patterns *must* exist within the memory map, and only the first 128 characters are defined in the BIOS ROM. Patterns for the remaining 128 characters are left to the programmer's design. A pointer in low memory must be set to point to a table of bit patterns that define the high 128 characters for use in graphics mode.

Two steps are required to create custom graphics characters. First, a table must be generated that defines the shape of each character. Second, the address of the first byte of the table must be stored in the BIOS data area in the external pointer interrupt location (label EXT_PTR at 7CH).

The PC displays graphics characters using an 8-by-8 pixel box. Each character graphic is defined with an eight-byte array; the first byte of the array specifies which pixels in the top row are illuminated, the second byte defines the pixels to be illuminated in the second row, and so on. Thus, a hollow box character would be defined as shown in the figure. The table of these arrays should be ordered in ascending byte value sequence; the first array defines the character pattern to represent the byte value 128 (80H), the second entry represents byte value 129, and so on.

DOS uses a load data segment (LDS) instruction to pick up the graphics character table address from location EXT_PTR. This instruction expects to find the segment offset followed by the segment address in four contiguous bytes. Therefore, the offset address of the graphics table must be stored in at EXT_PTR and the segment address in the next two bytes (offset address stored in bytes 7C and 7D, and segment address stored in bytes 7E and 7F).

The program below illustrates the technique by defining a hollow box graphic for a byte value of 128 (80H). To define the remaining characters, extend the array BOX (it also can be renamed) for as many characters as desired, allowing eight bytes for each character. A table containing patterns for 128 custom graphics characters would be 1,024 bytes—not long from the standpoint of program size—but designing 1,024 bytes worth of graphics characters is quite a job.

Allen DeLoach is a senior instructor in IBM's national marketing division located in Los Angeles. Mr. DeLoach holds a bachelor of arts in mathematics from Vanderbilt University.

FIGURE: *Hollow Box Character*

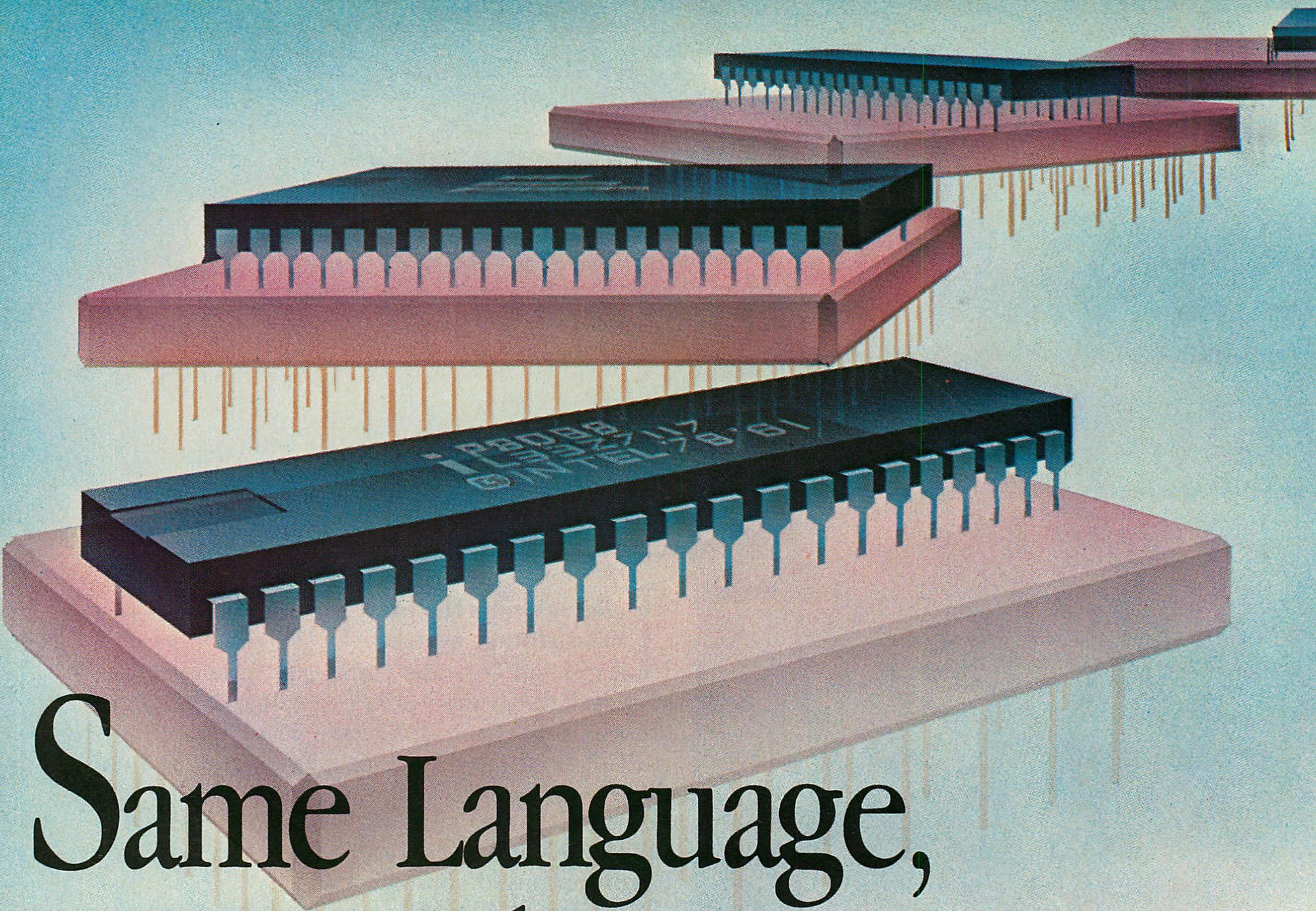
BYTE	BIT CONFIGURATION 7	6	5	4	3	2	1	0	HEX
0	1	1	1	1	1	1	1	1	FF
1	1							1	81
2	1							1	81
3	1							1	81
4	1							1	81
5	1							1	81
6	1							1	81
7	1	1	1	1	1	1	1	1	FF

Each character is defined with an eight-byte array; the first byte specifies which pixels in the top row are illuminated, the second byte defines the pixels in the second row, and so on.

LISTING: GRAFCHAR.PAS

```
program Grafchar;
const
                                   { 76543210 }
  BOX: array[0..7] of byte = ($FF, { 11111111 }
                              $81, { 1      1 }
                              $81, { 1      1 }
                              $81, { 1      1 }
                              $81, { 1      1 }
                              $81, { 1      1 }
                              $81, { 1      1 }
                              $FF); { 11111111 }

  BOX_byte: byte = $80;            {BOX byte value}
var
  BOX_char: char absolute BOX_byte;
  EXT_PTR_ofs: integer absolute $0000:$007C;
  EXT_PTR_seg: integer absolute $0000:$007E;
begin
  graphcolormode; palette(1);  {color; cyan-magenta-white}
  textcolor(1);                {cyan}
  gotoxy(1,2);
  writeln('Before EXT_PTR points to graphics table');
  gotoxy(1,4);
  writeln('Character value = ',Box_byte,' Graphic = ',BOX_char);

  EXT_PTR_ofs:=ofs(BOX);       {graphic table offset}
  EXT_PTR_seg:=seg(BOX);       {graphic table segment address}

  textcolor(2);                {magenta}
  gotoxy(1,8);
  writeln('After EXT_PTR points to graphics table');
  gotoxy(1,10);
  writeln('Character value = ',Box_byte,' Graphic = ',BOX_char);
  writeln; write('>>Press Enter: ');
  readln;
  textmode
end.
```


cheetah
Acinonyx jubatus
44-53 in. 86-143 lb
A SLEEK NEW STRATEGY FOR



Same Language, New Architecture

IBM and Microsoft have introduced versions of their macro assemblers with support for the 80186/286 and 8087/287; the new assemblers are compatible with the old architectures.

TED MIRECKI

Discussions about assembly language usually begin with apologies for its archaic features and end up with attempts to justify its supposed deficiencies in terms of its efficiency and the added control it provides over the hardware. Most experienced and sophisticated programmers, however, need no apologies or justifications; they are well-aware of the advantages of assembly language and well-versed in its use. For these programmers it is welcome news indeed when not one, but two new and improved versions of this vital programming tool become available.

IBM Macro Assembler 2.0 and Microsoft Macro Assembler 3.01 are full implementations, with all the capabilities needed for developing large and complex programs, such as operating systems, yet they are nimble enough for small, stand-alone utilities and modules that can be called from high-level languages. (The only differences that exist between Microsoft versions 3.0 and 3.01 are enhancements to the symbolic debugger included with the newer version. The assembler itself and all of its other utilities are the same.)

The big news in these two new versions is the support for the 80186 and 80286 processors and the 8087/80287 numeric coprocessors. Other enhancements include correction of sundry bugs, improved speed of assembly, additional assembly-time options, and various auxiliary programs, most notably a library manager. Otherwise, the capabilities as far as source language syntax, macros, structures, and conditional assembly are essentially un-

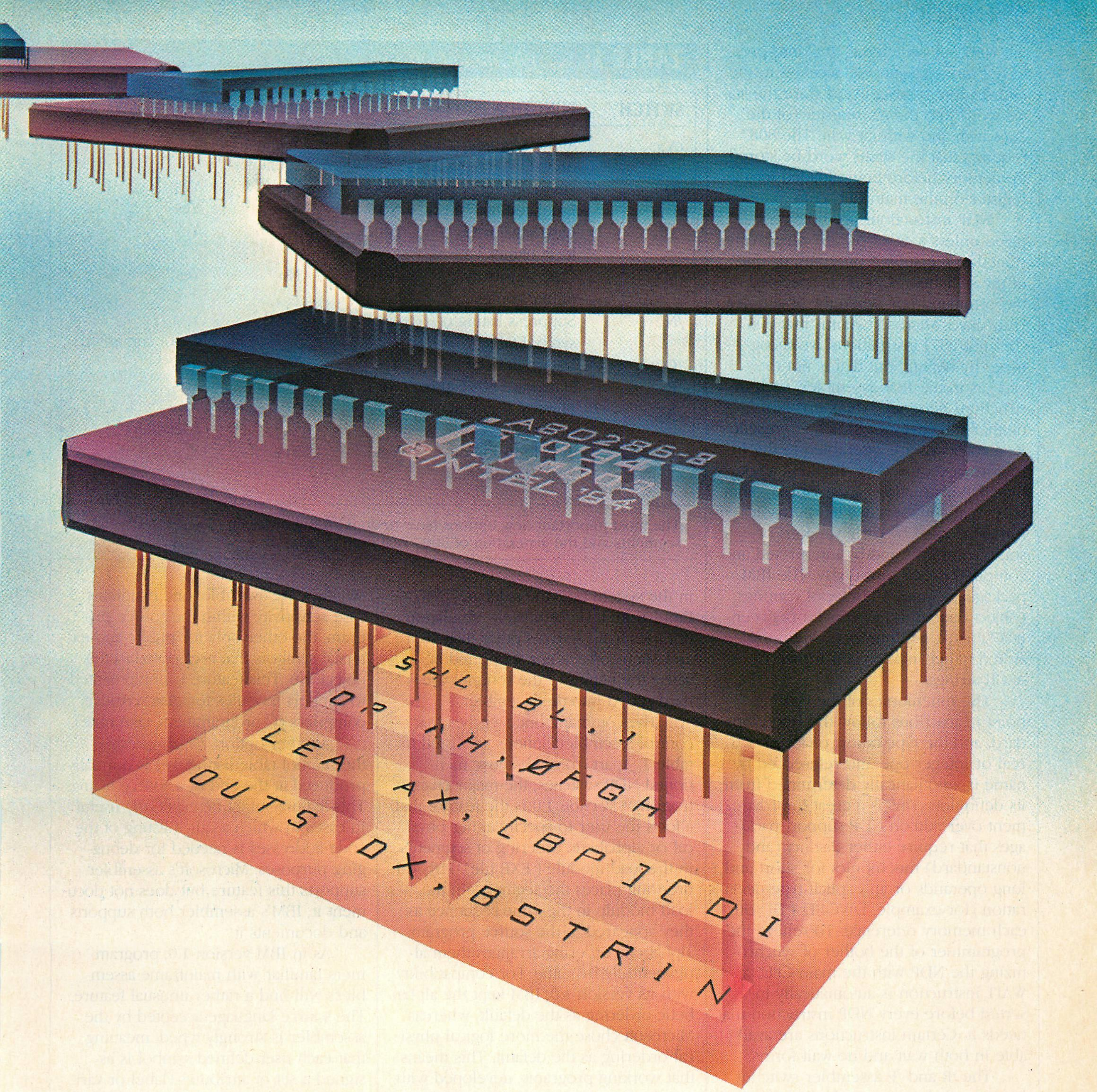


ILLUSTRATION • ALAN HASHIMOTO

changed from previous versions (specifically, IBM's MASM 1.0, which will be used as the basis of comparison).

The IBM and Microsoft assemblers are two versions of the same basic product; they are quite similar, but not identical. Throughout this article, references to *the assemblers* will mean that the feature being discussed is common to both the IBM and the Microsoft assemblers. Where differences exist, they will be explicitly noted.

The major difference is that Microsoft's product supports the 80286 processor in all modes, including the protected addressing mode that allows access to the full 16 megabytes addressable by this chip. IBM's assembler, on the other hand, supports the 286 only in real address mode, meaning that programs written for it can address no more than one megabyte of total memory organized in 64KB segments as for the 8088 and 80186. In effect, IBM provides support for the 186, not for the full capabilities of the 286, because the instruction set of the 286 in real address mode is identical to that of the 186. Support for the 186 is not mentioned in IBM's documentation because no IBM computers use this chip. The long-awaited operating system that allows use of the AT's extended address space has not been written in this new version of the IBM assembler.

The two assemblers implement essentially the same support for the NDP (Numeric Data Processor, or 8087/80287); the only difference is Microsoft's support of the full 287 instruction set, while IBM supports only that of the 8087. The 287 adds three instructions to

the 8087 set: switch the NDP into protected mode (to enable accesses to the entire address space), and store the status word into the AX register (of the 286) with and without wait. The 8087 requires that the status word be stored in memory before being loaded into a register on the main CPU chip.

NDP instructions are not recognized unless a .8087 or .287 directive is placed in the source program, or one of the switches, /R or /E, is specified on the command line when the assembler is invoked. Microsoft claims that 8087 (but not 287) instructions are recognized by default, but that is not the case: as in the IBM assembler, such instructions generate syntax errors if neither directive nor switch is present. Real constants are assembled in the Intel format required by the 8087 when the NDP instructions are enabled; otherwise, real constants are in Microsoft format as used by early versions of some compilers and by interpreted and compiled BASIC to this day. The IBM package includes a library of routines (object code only) to convert between 8087-format real numbers and numbers stored either in Microsoft format or as ASCII strings.

The mnemonics of the floating-point instructions follow the Intel standard, and the type (short or long) of a real or integer operand referenced by name is automatically determined from its definition. This is a great improvement over add-on NDP support packages that require either distinct (and nonstandard) mnemonics for short and long operands or an explicit type declaration (for example, DWORD PTR) on each memory reference. To relieve the programmer of the bother of synchronizing the NDP with the main CPU, a WAIT instruction is automatically inserted before every NDP instruction that needs it. Certain instructions are available in both wait and no-wait forms.

The /R and /E assembler switches mentioned above control which kind of object code is generated for NDP instructions. /R produces real hardware instructions for the NDP, so that the resulting code can be run only on a system that actually has an NDP installed. /E produces calls to routines that emulate the NDP operations in software. The emulation library is not provided with the assembler, but comes with the IBM/Microsoft FORTRAN and Pascal compilers. Typically, such libraries implement the math functions in both hardware and software and determine at runtime which to execute depending on the presence or absence of an NDP in the system (see the sidebar, "8087 Emulation"). Unfortunately, emulation libraries from sources other than IBM and Microsoft are not supported because the names of the routines are hard-coded into the assembler.

TABLE 1: *Assembly-time Options*

SWITCH	FUNCTION	IBM 1.0	IBM 3.0	MICROSOFT 3.01
/A	Alpha segment order	No	Yes (default)	Yes
/D	Pass 1 listing	Yes	Yes	Yes
/E	Emulated 8087 instr.	No	Yes	Yes
/ML	Case sensitivity, local names	No	No	Yes
/MX	Case sensitivity, public names	No	No	Yes
/N	Suppress listing of symbol table	No	Yes	Yes (undocumented)
/O	Listing in octal	Yes	Yes	Yes
/R	Hardware 8087 instr.	No	Yes	Yes
/S	Physical segment order	No	Yes	No (default)
/X	Toggle listing of false conditionals	Yes	Yes	Yes

The most important new options have to do with the ordering of generated segments and the generation of 8087 real number support.

The command-line switches that control assembler options are listed in table 1. Apart from the above-mentioned NDP switches, the major change from IBM version 1.0 is the feature that allows the user to specify either physical or alphabetic ordering of segments in the load module (.EXE file). Physical ordering places the segments in the load module in the same sequence as they appeared in the source program; alphabetic ordering arranges them alphabetically by name. For compatibility with its version 1.0, IBM kept the alphabetic ordering as the default, whereas Microsoft chose the more logical physical ordering as the default. This means that working programs, developed with version 1.0 and whose operation depends on a particular segment order, might mysteriously hang the system when run after reassembling without using the /A switch on Microsoft 3.01.

Other new command-line options, for the Microsoft assembler only, include case sensitivity for source symbols. Normally, the assembler converts all alphabetic input to uppercase, so that, for example, "LABEL1", "Label1" and "label1" are considered the same symbol. If case sensitivity is enabled, alpha input is left in whatever case was entered by the programmer, so these three symbols would be distinct. Two switches are available: one maintains case sensitivity within the source program, the other only for public and external symbols that are copied to the object file. This feature is implemented to support modules for case-sensitive compilers (C programmers, take note).

Another option, /N, suppresses the symbol table listing that is normally produced at the end of a source listing. This feature might be especially useful in cases in which a quick listing of the code addresses is needed for debugging purposes. Microsoft's assembler supports this feature but does not document it; IBM's assembler both supports and documents it.

As in IBM version 1.0, programmers familiar with mainframe assemblers will find a rather unusual feature. The source language accepted by the assembler is strongly typed, meaning that each user-defined symbol is assigned a set of attributes (label or variable, byte or word, near or far) that determines the legal uses of that symbol. Strong typing is common for high-level languages, where the objective is to protect the programmer from his own mistakes. It is rare in an assembler, where it is normally assumed that the programmer should be allowed to do whatever he pleases. Typing has been strengthened in the new versions. In version 1.0, the moving of data to or from a register implied a word or byte move depending on the size of the register, not the type declared for the data. For example, the instruction MOV AX,VALUE fetched a word from the ad-

dress at which the symbol VALUE was defined, regardless of the type of data that was defined by that symbol, because the AX register operates on words only. In the new versions, the type of VALUE had better be WORD or the assembler will complain about mismatched types. Alternatively, the symbol may be cast into the proper type by use of the WORD PTR operator. On the one hand, this could be a nuisance, but on the other, it may catch errors of the type where AX was mistyped for AH.

The stronger typing is supported by a new operator, .TYPE. This is not to be confused with the TYPE operator (no leading period) that was implemented in version 1.0 and that returned the length in bytes of a data variable or an indication of whether a label was near (in the same segment) or far (in a different segment). The new .TYPE operator returns a byte whose bits indicate the various attributes of the symbol operated on: defined or undefined, absolute or relocatable, variable or label, local or public. This is a great feature for implementing macros that must generate different code depending on the attributes of the supplied parameters.

The other operators used in forming operand expressions are basically unchanged from version 1.0. In the first version, however, some of the operators were undocumented, and many logical and arithmetic expressions worked unreliably or not at all. Although it is difficult to perform an exhaustive test of all possible expressions, no obvious bugs were discovered in any of the tests that were performed.

In general, both of these new assemblers are quite a bit more reliable than version 1.0, and several outright bugs have been corrected. Packed decimal and real constants are generated properly, and labels outside of segments are flagged as errors. The problem reported in Tech Notebook 38 ("A Bona Fide Bug," Ted Mirecki, *PC Tech Journal*, May 1985, p. 41) turns out to have been a bug in the version 1.0 assembler, not the DOS 2.x linker as was reported. With the version 1.0 assembler, a program containing uninitialized DD, DQ, or DT fields within structures would not link with any linker more recent than the one supplied with DOS 1.1. When run through the new versions of the assembler, the same source code produces object files that are acceptable to any version of LINK.

It is therefore not surprising that, for the same source program, the object file produced by either new version of the assembler is different from the object file produced by version 1.0. What is surprising is that the .EXE files produced from these different object files, by the same LINK program, are also different. The contents of memory after the different .EXE files are loaded are identical, as are the .COM files produced by EXE2BIN from the output of any assembler version. Once a working program that was developed with version 1.0 is successfully assembled with a new version and linked, it will operate as before, because its memory image remains unchanged.

The listing output produced by the new versions is the same as that from version 1.0. Creating a cross-reference listing still requires a separate program

Strong typing is common for high-level languages, but it is rare in an assembler, where it is normally assumed that the programmer should be allowed to do whatever he pleases.

(the CREF utility is provided with both assemblers), and the formats of listings with and without cross-referencing are different (a line number is appended to each source line only if cross-referencing is requested). This makes writing general utilities that process listing files (see listing 1) difficult. One improvement has been made over version 1.0: a PAGE directive that changes the number of lines per page now works properly. In the old version, the first page after the line count was increased contained only as many lines as the difference between the new line count and the default line count of 57.

A noteworthy feature of the listing format is that the object code displayed at the left of each source line differs in several respects from the actual machine code. Word values (for example, offsets of data in memory and 16-bit immediate operands) are displayed in the listing in high-byte/low-byte order as they appear when loaded into registers, whereas in memory they are stored in low-byte/high-byte order. The targets of jump and call instructions are identified in the listing by offsets from the start of their segments, but in the machine code they are represented as offsets from the instruction following the jump or call. This representation makes finding the target location in the listing easy, because the offset of each instruction from the start of the segment is listed along with the object code. Compare this with the ROM BIOS listings in IBM's *Technical Reference Manual* (produced with Intel's ASM86), where the object code is a true memory image. To find the target of a jump or call in the BIOS listing, the last two bytes of the instruction must be reversed and that word value added to the address of the following instruction.

Purists might complain that the object code listing should faithfully reproduce the actual machine code—that is, the contents of memory at runtime. This was certainly true in the old days of batch processing on mainframes, where the primary debugging tool was a hard-copy memory dump. To see what this looked like, display some executable code with DEBUG's D (Dump) command. Without a verbatim listing of object code, the program's instructions could never be found in such a mess. But today's interactive debugging tools make reading dumps obsolete. Even DEBUG, provided free with DOS, has a U (Unassemble) command that automatically translates machine code to a form that is intermediate between the source format and the object code listed by the assembler. All in all, Microsoft's implementation of the assembly listing is more useful than Intel's.

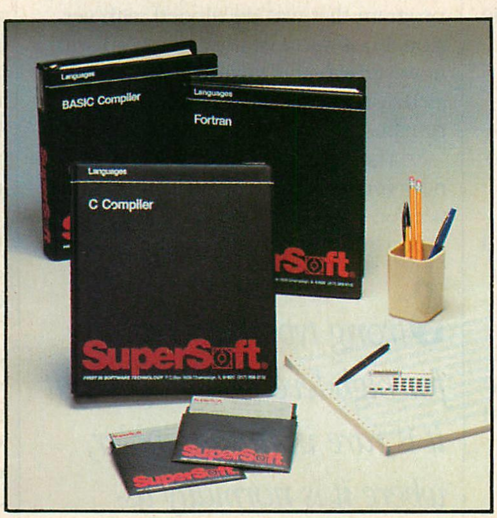
One feature that the two new assemblers should have, but do not, is the ability to invoke macros and conditional assembly statements within structures. Using an IRP or REPT macro to generate a structure containing a set of fields with unique names, instead of enumerating each one, would be convenient. Perhaps one of several structures differing in only one field must be generated. Although the different structures can appear within if-then-else constructs, that requires repeating the nonvarying portion for each condition. The assembly source modules provided with the Lattice C compiler illustrate the problems caused by this restriction.

EASE OF USE

Installation of the new assemblers involves nothing more than copying the required files to a DOS disk. A useful working system consisting of the assembler, cross-reference utility, linker, library manager, editor, and a subset of the DOS external commands fits on one floppy disk, so that the development cy-

SuperSoft Languages
For Performance and Service

For seven years the name SuperSoft has been trusted by professional software developers throughout the world. Trusted because SuperSoft means quality. It means performance. It means an unshakable dedication to customer service. It means that when you call SuperSoft with a problem, you get a trained technician ready to help. That's service you can depend on—and it comes with every product we sell. In addition, if you need software with special features, we can customize our products to fit your needs.



SuperSoft BASIC 3.0

The Interpreter Compatible Compiler

Now you can use the superior BASIC compiler for improved performance. SUPERSOFT BASIC. Compatible with the most commonly used interpreter, SuperSoft BASIC is the only compiler on the IBM PC which meets or exceeds the ANSI X3.60 – 1978 BASIC Standard. It comes complete with:

- **Full 8087 Support**—for lightning fast math operations
- **IEEE floating point**—for fast, precise scientific calculations
- **BCD math**—for exact decimal numbers in business applications
- **Fully implemented double precision**—for unbeatable math accuracy
- **Precise error messages/codes and sub-codes**—for simplified debugging

SuperSoft BASIC is also highly compatible with the BASIC interpreter sold by both Microsoft and IBM.

And, to top it off, SuperSoft BASIC:

- Allows easy interface to C and Assembly code
- Provides fully ROMable output
- Allows dynamic loop nesting
- Allows 32,767 bytes per record and 65,535 records per random file.
- **And requires no run-time license fee for distribution of your compiled code.**

SuperSoft BASIC (PC-DOS, MS-DOS, CP/M-86): $395

SuperSoft C

SuperSoft's C compilers and C cross-compilers are designed to meet the highest standards of quality, reliability, and performance.

- **Over 200 library functions provided in source code form**—for full control over your libraries
- **Full line of compilers and cross-compilers**—for use with the Z80, 8080, 8086, 8088, Z8001, Z8002, and others
- **Fully overlay support**—for extremely large application programs
- **Inline assembly code interface**—for easy insertion of assembly code
- **BCD math**—for full decimal accuracy
- **Fully ROMable assembly code output**—for applications which must be burned into ROM memory
- **General compatibility with UNIX version 7 C**—for standard C programming

Available on the most popular operating systems, these features make SuperSoft C your best choice for a professional quality compiler.

SuperSoft C (PC DOS, MS-DOS, CP/M-80, CP/M-86): $395
SuperSoft C Cross-Compilers: $1,000

SuperSoft FORTRAN

For years, serious developers have been using SuperSoft FORTRAN to bring their large applications down to microcomputers. Why? Because SuperSoft FORTRAN offers major advantages over other FORTRAN compilers.

- **For compiling extremely large programs**—extended code and data space under (PC & MS-DOS) plus overlay support
- **For faster math operations**—optional 8087 support
- **For mainframe compatibility**—full ANSI '66 FORTRAN
- **For structured programming techniques**—optional RATFOR preprocessor
- **For those without unlimited memory**—only 128K of memory required
- **For unbeatable math accuracy**—double precision and complex numbers using IEEE floating point
- **For fast, compact code**—buy SuperSoft FORTRAN

SuperSoft FORTRAN (PC DOS, MS-DOS, CP/M-80, CP/M-86): $395

To Order, Call: 800-762-6629

In Illinois call: 217-359-2112

SuperSoft FORTRAN was developed by Small Systems Services, Urbana, IL

Japanese Distributor: ASR Corporation International, TBL Building, 7th Floor, 1-19-9 Toranomon, Minato-Ku, Tokyo 105, Japan Tel. 03-05025550 Telex 222-5850 ASRTYO J

Spain: Ibersoft, Vicent Jove 54, Apartado 336, Gijon, Oviedo, Spain telex 87330
Netherlands and Belgium: Softkey, Gedempte Gracht 4, 7400 GX Deventer, the Netherlands telex 49540

SuperSoft

SuperSoft, Inc., P.O. BOX 1628
Champaign, IL 61820 Telex 270365

CIRCLE NO. 178 ON READER SERVICE CARD

cle of editing through testing can be performed with no disk swaps on a two-drive system. Both manuals provide an inventory of the supplied files and offer suggestions on distributing them over two disks to allow accesss to other utilities (more on these later).

All input and output files may be specified by full path names. Path support could be improved by a provision to specify a series of paths to be searched for include files; as it is, the location of include files in the directory structure must be explicitly coded in the source program. Include files may be nested to any level, provided that a sufficiently high number of files has been specified for the FILES parameter in the CONFIG.SYS file.

Error messages are essentially unchanged from version 1.0. They are good but not great: the all-inclusive and uninformative "Syntax error" message appears all too often. A most useful addition is the return of a system error code upon assembly termination: 1 if any errors were detected, 0 if none. This may be tested by an IF ERRORLEVEL statement in a batch file to skip subsequent cross-reference and link steps when the assembly fails.

As mentioned above, type checking is more rigidly applied than it was in version 1.0. Only variables defined with DW or forced to words with WORD PTR may be moved into or out of 16-bit registers, and only byte values may be transferred to or from 8-bit registers. The previous version was more forgiving, and some programs that had assembled cleanly may produce errors with the new versions. Another formerly acceptable construct that is now flagged as a syntax error is an operand of the type SS+CONST.FF, where SS is a label at which a structure is invoked, CONST is an absolute symbol or constant, and FF is a field within the structure. This must now be written as SS.FF+CONST. All the new errors are adequately identified, and a clean assembly of an old program should require no more than two or three passes through the new version.

Timing the execution of code produced by different assemblers is a pointless exercise, because they all must generate identical machine code for a given source file. The primary measure of assembler performance is the speed of assembly. The results for several programs of various sizes are listed in table 2. The longest widely available assembly program is VDISK.SYS, provided in listing form with DOS 3.1. To convert the listing to source code suitable for the assembler, process it with the BASIC program reproduced in listing 1.

The speed in generating macros is tested by the program MACTEST.ASM (listing 2). This is not meant to be an executable program, nor is the implementation of the if-then-else macros practical; the objective here was merely to time the maximum number of nested macro invocations.

As compared to IBM version 1.0, the results for the new assemblers are impressive, yielding an overall speed improvement of about 2 to 1. The improvement is especially noticeable in the macro test, where the advantage is almost 3 to 1. The differences between IBM and Microsoft are negligible. With either one, loading the assembler from a floppy disk often takes longer than processing a short program.

TABLE 2: *Assembly Time*

	SOURCE SIZE	IBM 1.0	IBM 2.0	MICROSOFT 3.01
BOOT.ASM[1]	4KB, 140 lines	0:13	0:10	0:10
Speed ratio		1.0	1.3	1.3
PLOT.ASM[2]	7KB, 280 lines	0:22	0:15	0:15
Speed ratio		1.0	1.5	1.5
PACKDIR.ASM[3]	20KB, 480 lines	1:06	0:36	0:37
Speed ratio		1.0	1.8	1.8
VDISK.ASM[4]	74KB, 2,150 lines	4:55	2:13	2:16
Speed ratio		1.0	2.2	2.2
MACTEST.ASM[5]	N/A			
1 iteration		0:28	0:15	0:15
Speed ratio		1.0	1.9	1.9
10 iterations		4:02	1:26	1:29
Speed ratio		1.0	2.8	2.7

[1] *"Customized Boots," Michael Abrash and Dan Illowsky,* PC Tech Journal, *February 1985, p. 150.*
[2] *"Plotting Data," Peter G. Aitken,* PC Tech Journal, *March 1985, p. 123.*
[3] *"Dipping Into Directories," Ted Mirecki,* PC Tech Journal, *February 1985, p. 67.*
[4] *Supplied with DOS 3.0; see text.*
[5] *See listing 1.*

All times are given in min:sec.
Assembly times are for assembler and object files on RAMdisk, source code on floppy disk.

The assembly times given are applicable when the assembler and object code file are on RAMdisk and the source code file is on floppy disk.

AUXILIARY PROGRAMS

Both assemblers provide a cross-reference utility, a linker, and a library manager. The CREF program produces a listing file that specifies the source line numbers where each symbol was used. The need to perform an additional step to generate this is a minor annoyance. More annoying is the fact that the source listing does not contain line numbers unless a cross-reference is requested. IBM and Microsoft provide different versions of CREF, but the two are functionally identical.

IBM includes version 2.2 of the LINK program, the same version supplied with DOS 3.0. Microsoft provides version 3.0 of LINK, a newer version that implements some additional options: the maximum number of segments the linker can process may be increased from the default of 128, case sensitivity in public names can be maintained, and some control is provided over the ordering of segments in the output file. Both linkers can handle object libraries with various page sizes (this feature is explained below).

The library manager is a welcome addition to an assembler package. Assembly language provides none of the amenities of high-level languages (like formatting of numbers), so programmers usually develop a toolkit of well-tested routines that can be incorporated into other programs. Without a library manager, all the routines required by a program must be enumerated at link time. A library manager allows these routines to be put into a library from which the linker can automatically extract the ones needed by a particular program. Both assemblers provide the Microsoft LIB program with the capability to create libraries from object files, add, delete, and replace object modules, and produce cross-references of public symbols in the library.

A new feature of this version of LIB is the ability to vary the page size. Each object module is allocated an integral

number of pages in the library file; a smaller page reduces the size of the library, a larger one reduces link time. The default page size is 512 bytes for the IBM version of LIB, and 16 bytes for the one with the Microsoft assembler.

The variable page size solves one incompatibility between various versions of Microsoft systems software. The LINK program supplied with DOS 2.x and with compilers prior to versions 3.0 of FORTRAN and Pascal could process library files with a page size of 512 bytes only. Later versions of these compilers had libraries with different page sizes that could be used only with the version of the linker with which they were supplied. The new LIB can convert libraries to a different page size.

IBM includes an interesting program called SALUT, or Structured Assembly Language Utility, which is a preprocessor that accepts modified assembly language as input and produces standard assembly language as output. The language extensions are a set of structured programming constructs, such as if-then-else, do-while, and do-until. These are inserted into the source after standard compare or test instructions, in place of loop and conditional jump instructions. The resulting program is a hybrid between assembly and a high-level language. The difference between SALUT and conditional assembly is that the former generates the branching instructions executed at runtime, while the latter performs tests and branches at assembly time.

Another extra program in the IBM package is ASM, or the small assembler. It is meant for systems with less than 128KB and does not implement macros, structures, or conditional assembly. Because it requires at least 96KB of memory, its usefulness is very limited.

The Microsoft assembler includes a symbolic debugger with somewhat more capability than the standard DOS DEBUG. But it is more useful with high-level languages than with an assembler: its primary claim to fame is that it can display the source lines corresponding to a series of machine instructions. This works for IBM/Microsoft FORTRAN and Pascal and Microsoft/Lattice C, but not for assembly language, because the assembler does not insert source line numbers into the object file. This would be of marginal utility anyway, because disassembly already produces output very close to the source format of assembly language. One advantage of this debugger over the standard is that it can refer (for purposes of display or setting of breakpoints) to public symbols by name instead of by address. Although assembly language programs do not typically contain many public symbols, the programmer may define extra ones just for purposes of debugging.

This debugger has several other advantages over DOS DEBUG. First, DOS commands (both internal and external) can be performed during a debugging session. Second, input to the debugger or to the program being debugged can be redirected to a disk file or device at any time, not only at startup, and can be returned to the keyboard. Similarly, output of either the debugger or the program can be temporarily redirected and then retuned to the screen. Finally, numeric data can be displayed in real or integer decimal formats, as well as hexadecimal.

The MAKE utility, derived from UNIX, is a more useful program offered with the Microsoft assembler. It provides a way to automate the rebuilding of one or more programs whenever a component of the system is updated. The process is controlled by a file that specifies the sequence of operations (assemblies, compilations, library updates, links) required to build an application, as well as which output files are created from which input files. For example, .EXE and .COM files are created from object files and libraries, libraries from object files, and object files from source and include files. When MAKE is executed, it tests the dates of all the components and performs only those operations for which the input files are more recent than the outputs. For this process to work, the system's clock and calendar always must be set properly. The usefulness of the MAKE utility is not limited to the assembler. It is a great help in maintaining a complex applications system in any language.

The Microsoft and IBM assemblers are no exception to the rule that no one should expect to become an expert solely from reading the assembler manuals. In fact, most assemblers are unusable without some additional documentation on the hardware characteristics of the particular processor.

The Microsoft manual admits this at the outset; the user is warned of the need to know the operation and instruction set of the 8086/186/286 family of microprocessors, and is referred to "one of the many books that define them;" however, no specific titles are mentioned. The single manual is divided into two sections: a user's guide

8087 EMULATION

Certain high-level languages have the capability of automatically switching between executing hardware 8087 instructions and emulating them in software, depending on the presence or absence of the NDP at runtime. The IBM and Microsoft assemblers provide this same capability for routines that are to be called from IBM/Microsoft FORTRAN and Pascal.

The sequence of 8087 instructions shown in figure 1 will be used as an example. This routine does nothing useful and is not meant to be executed; rather it is meant to illustrate the emulator's encoding of some instructions arbitrarily chosen from the 8087's instruction set.

Figure 2 is a disassembly (produced by DEBUG) of the code assembled with the /E option and linked without the emulator library. Note the WAIT instructions inserted by the assembler before every NDP instruction except FINIT; these WAITs are inserted regardless of the /E switch. In fact, if the /E option had not been used, the only visible difference in the code would be the absence of the NOP before the second-to-last WAIT. The output of the assembler with and without the /E switch is functionally identical, and in either case the code is perfectly acceptable to the 8087.

One other effect of the /E option is to insert into the object file references to the emulator routines and pointers to the locations where calls to these emulator routines need to be inserted. The actual patching of the code is performed by the linker only if the emulator library is available at link time. Therefore, an object file assembled with the /E option can be used with or without emulation, depending upon the parameters supplied to the linker. The "Unresolved external reference" errors generated when linking without the emulator library may be ignored, because the calls to these routines are not present.

Figure 3 is a disassembly of the same object file after linking it with an emulator library (MATH.LIB supplied

FIGURE 1: *Sequence of 8087 Instructions*

```
        .8087
CODE    SEGMENT BYTE 'CODE'
        ASSUME  CS:CODE, DS:CODE

START:  FNINIT                  ;INITIALIZE, NO WAIT
        FINIT                   ;INITIALIZE WITH WAIT
        FLD     REAL_4          ;LOAD SHORT REAL
        FLD     REAL_8          ;LOAD LONG REAL
        FADD                    ;ADD
        FIST    INT_2           ;STORE 16-BIT INTEGER
        FIST    DWORD PTR[BX]   ;STORE 32-BIT INTEGER
        FSTSW   STAT_WORD       ;STORE STATUS WORD
        FWAIT                   ;WAIT FOR 8087 READY
        WAIT                    ;DO NOT USE WITH EMULATOR!

REAL_4    DD    ?
REAL_8    DQ    ?
INT_2     DW    ?
STAT_WORD DW    ?
CODE      ENDS
          END   START
0000C 062600    DB      06,26,00
000F CD3A       INT     3A
0011 C1         DB      C1
0012 CD3B       INT     3B
0014 162E00     DB      16,2E,00
0017 CD37       INT     37
0019 17         DB      17
001A CD39       INT     39
001C 3E3000     DB      3E,30,00
001F CD3D       INT     3D
0021 9B         WAIT
```

This is the source code for the disassembled listings that are shown in figures 2 and 3 at right.

FIGURE 2: *Linked without Emulator Library*

```
0000 DBE3       FINIT
0002 9B         WAIT
0003 DBE3       FINIT
0005 9B         WAIT
0006 D9062200   FLD     DWORD PTR [0022]
000A 9B         WAIT
000B DD062600   FLD     QWORD PTR [0026]
000F 9B         WAIT
0010 DEC1       FADDP   ST(1),ST
0012 9B         WAIT
0013 DF162E00   FIST    WORD PTR [002E]
0017 9B         WAIT
0018 DB17       FIST    DWORD PTR [BX]
001A 9B         WAIT
001B DD3E3000   FSTSW   [0030]
001F 90         NOP
0020 9B         WAIT
0021 9B         WAIT
```

This is the disassembly of the code in figure 1, assembled with /E, but linked without the emulator library.

FIGURE 3: *Linked with Emulator Library*

```
0000 DBE3       FINIT
0002 CD37       INT     37
0004 E3         DB      E3
0005 CD35       INT     35
0007 062200     DB      06,22,00
000A CD39       INT     39
```

This disassembled code was assembled with the /E option and then linked with the emulator library, MATH.LIB.

with the Microsoft Pascal compiler, version 3.2). The disassembly was produced by hand, because DEBUG considers the data bytes interspersed among the instructions as executable and attempts to disassemble them. An important point to note is that only two 8087 instructions survive. One is the no-wait FINIT, which is executable in the absence of an 8087. The other is the last instruction in the sequence, which in the source was coded as WAIT instead of FWAIT. If this instruction were executed on a system that did not contain an 8087, the system would wait forever for the absent 8087 to indicate that it is ready. WAIT and FWAIT generate the same object code, but the difference between them is that the latter is replaced by the emulation routines. That is why FWAITs should be coded in place of WAITs in any program that might be run by an emulator. All other NDP instructions are replaced with interrupts that point to the emulator routines.

Each interrupt instruction, which is two bytes long, replaces an assembler-generated WAIT and the first byte of the 8087 instruction. Following this are one or more bytes of data representing the remainder of the NDP instruction. The information from the first instruction byte is carried in the interrupt number itself. Therefore, the two FIST instructions are represented by different interrupt numbers, because the first byte in each form of FIST is different (DF and DB). On the other hand, the FINIT instruction and the second FIST are handled by the same interrupt because they both begin with DB. The encoding of FWAIT, which is an 8088 rather than an 8087 instruction, is an exception. Exactly what the emulator needs to do in order to simulate a WAIT is not entirely clear; Intel's documentation suggests that FWAITS be replaced with NOPs.

The entire set of 8087 instructions uses interrupts in the range 34H through 3EH. In the *Technical Reference Manual,* these are marked as reserved for DOS, and if an emulator package is not being used, they all point to one location within DOS that contains an IRET (return from interrupt) instruction. A Pascal or FORTRAN program, during its initialization, repoints each of these interrupt vectors to the appropriate emulator routine within its load module. Therefore, the emulator package is not usable with programs written totally in assembly language.

At runtime, when an emulation routine gains control with an interrupt, it checks for the existence of the NDP. If an NDP does not exist, the routine performs the floating-point operation with software, using the data bytes following the interrupt to determine the type of operation and the address of the operands. Then it adjusts the return address to point past the data and returns to the next instruction in the calling program.

If, however, an NDP is present, the emulation routine replaces the interrupt instruction with a WAIT and the first byte of the original floating-point instruction, restoring the code to its original state after assembly. The return address is adjusted downward by two bytes, and the return is to the restored NDP instruction, which is then executed. If the flow of execution should subsequently return to this instruction (as in a loop, for example), the emulation routine is not invoked again because the interrupt is no longer there.

—TM

AZTEC

Manx Aztec C86 is the best C for MS-DOS and you can prove it yourself!

"A compiler that has many strengths ... quite valuable for serious work"
Computer Language review, February 1985

Manx Aztec C86 - The C For MS-DOS

Manx Aztec C86 is clearly the best C software development system for MS-DOS. Aztec C86 is the only C compiler for MS-DOS that provides the level of performance, features, documentation, and support required for serious, professional software development. You can prove it yourself. All you have to do is order Aztec C86 from Manx, evaluate it, and, if you like it, keep it. If you don't like it, send it back within 30 days and we'll cancel your order.

If you keep your Manx Aztec C86, as 99% do, you'll be in with the best company.

Manx Aztec C86 Features:

Optimized C compiler: Unsurpassed for code quality and speed. Optionally generates 80186 and 80286 code. Full K & R.

Symbolic Debugger: Execution trace, break points, display data in floating point, integer, character, or hex format. Evaluate expressions. Detect illegal memory stores, modify memory/registers, disassemble code.

Manx AS86 Macro Assembler: Supports macroes, 8086, 80186, and 80286 instructions in Intel format. Fast execution.

LN86 Overlay Linker: Links small, large, and mixed memory model routines, supports overlays, and options for producing ROM based code.

Librarian: Build and modify personal or system run time libraries.

8087/80287 Sensing Library: One library simulates floating point, another assumes the presence of an 8086 or 80287 math chip, the third senses the existence of a math chip, and if it finds one it uses it.

Profiler: Provides a run time analysis of your code to pinpoint code segments to optimize.

UNIX Library: Compatible with UNIX C. Fast I/O. Terminal I/O can be buffered or unbuffered.

DOS Library: Time and date functions, program forking (exec), program chaining, directory commands, I/O port support, sysint support, BIOS functions, and BDOS functions.

Screen & Graphics Library: Screen and cursor functions. Fast routines for drawing lines, circles, elipses, points, and setting colors.

CP/M-86 Library (-c): Produce programs for CP/M-86.

Large Memory Model: Manx Aztec C86 supports programs and data of any size. Global data has a max size of 64k.

Intel Object Option: Interface to software that requires Intel object format, such as PLINK86.

Z (vi) Source Editor (-c): Fast, powerful editor, Macro capabilities, undo, ctags, buffers for commands and data, and all the bells and whistles that make vi fanatics fanatical.

ROM Support Package (-c): Startup routine, linker options for separate placement of code and data, special utilities like the Intel HEX Utility, documentation, and library source.

Library Source Code (-c): UNIX, screen, graphics, and math function libraries.

Mixed Memory Models (-c): Mix large code and small data, small code and large data, or mix within type.

UniTools (-c): The UNIX utilities make, diff, and grep.

One year of updates (-c): As new versions are released, updates are automatically sent.

Technical Support: Manx has a full time staff to provide support via telephone & bulletin board.

Items marked -c are special features of the Aztec C86-c system.

Manx Aztec C86 is available in four configurations: Manx Aztec C86-c, Manx Aztec C86-d, Manx Aztec C86-p, and Manx Aztec C86-a. The -p and -a systems are not intended for commercial work and do not incorporate the same compilers as the -c and -d systems. All systems are upgradable.

Aztec C86-c Commercial System	**$499**
Aztec C86-d Developer's System	**$299**
Aztec C86-p Personal System	**$199**
Aztec C86-a Apprentice System	**$ 49**

Manx Cross Development Systems

Manx Aztec C compilers are available as native or as cross development systems for PC-DOS, MS-DOS, Macintosh, CP/M-86, CP/M-80, TRSDOS, Apple II, and Commodore 64/128.

Cross development involves two computer systems: the development system (HOST) and the execution system (TARGET). This method is useful when the TARGET machine is slower or more limited than the HOST.

HOSTS: VAX UNIX ($3000), PDP-11 UNIX ($2000), MS-DOS ($750), CP/M ($750), Macintosh ($750), CP/M-68k ($750), XENIX ($750).

TARGETS: MS-DOS, CP/M-86, Macintosh, CP/M-68k, CP/M-80, TRS-80 3 & 4, Apple II, Commodore C64, 8086/80x85 ROM, 68xxx ROM, 8080/8085/Z80 ROM, 65xx ROM.

Additional TARGETS are $300 to $500 (non VAX) or $1000 (VAX). Call for information, on cross development to the 68000, 65816, Amiga, C128, CP/M-68K, VRTX, and others.

How To Become a Manx Aztec C User

Call 1-800-221-0440 or 1-800-832-9273 (800-TEC WARE). In NJ or outside the USA call 201-530-7997. Orders can also be telexed to 4995812.

Payment can be by check, COD, American Express, VISA, Master Card, or Net 30 to qualified customers.

Orders can also be mailed to Manx Software Systems, Box 55, Shrewsbury, NJ 07701.

For More Information: call 1-800-221-0440, or 201-530-7997, or write to Manx Software Systems.

Manx maintains a large professional staff to service and support Manx users. You will get fast delivery and great service dealing directly with Manx.

Support Software for Manx Aztec C86

C-tree $395: B-tree database system. Easy to use. Available for Aztec C for MS-DOS, Macintosh, CP/M-86, CP/M-80, and others. Includes source.

PHACT $250: Powerful database system. Available for Manx Aztec C compilers for MS-DOS, CP/M-86, CP/M-80, and Macintosh.

PANEL $295: Create screens via simple, powerful editing commands. Select colors, edit fields. Directly input data to a multi-keyed file utility included with the system.

SunScreen $99: Create and modify formatted screens easily. Validate fields, select colors, create screens for both the color and monochrome cards. With library source SunScreen is $199.

WindScreen $149: Combines SunScreen with a powerful window utility.

Windows for C $195: Versatile window utility that supports IBM PC compatible and some non-compatible environments.

AMBER Windows $99: Powerful, low priced window package.

HALO $250: The ultimate C graphics package. It supports viewports, shapes, and multiple graphics cards. A less expensive version is available for just the PC mono and color cards.

FirsTime $295: Syntax checking while you edit greatly shortens compile time.

Pre-C $395: Powerful Lint-like utility locates structural and usage errors. Easily checks multiple files for bad parameter declarations and other interface errors. Lint users will find the user interface a dream come true.

PC-LINT $98: Lint-like utility that supports large memory models, has clear error messages, and executes quickly, has lots of options and features that you wouldn't expect at this low price.

Greenleaf Functions $185: Source for over 200 C and assembler functions. They are great, they work, they are used extensively, and are economically priced. Clear documentation and easy to use interface round out an impressive package.

C Utility Library $185: C and assembler source for screens, windows, color graphics, asynch communications, and more. The color graphics and speed of this package are impressive.

Plink-86 $395: MS-DOS linkage editor for producing and maintaining overlayed programs. It works with Aztec C86 in Intel object format mode.

30 Day Guaranty:

Any Manx Aztec development system can be returned within 30 days for a refund if it fails to meet your needs. Restrictions are that the original purchase must be directly from Manx, shipped within the USA, and the package must be in new condition. Returned items must be received by Manx within 30 days. A restocking fee may be required.

Discounts:

There are special discounts available to professors, students, and consultants. A discount is also available on a "trade in" basis for users of competing C systems.

Manx Aztec C Distribution:

In the USA, Manx Software Systems is the exclusive distributor of Aztec C. Telephone or mail order sales other than through Manx are unauthorized.

MANX

To order or for information call:

800-221-0440

UNIX is a registered TM of Bell Laboratories, Lattice TM Lattice Inc., C-tree TM Faircom, Inc., PHACT TM PHACT ASSOC., CI Optimizing C86 TM Computer Innovations, Inc., MACINTOSH, APPLE TM APPLE, INC., Pre-C, PLINK 86, TM PHOENIX, HALO TM Media Cybernetics, Inc., C-terp, PC-lint TM GIMPLE Software, Windscreen, SunScreen TM SurfSc, PANEL TM Roundhill Computer Systems Ltd., WINDOWS FOR C TM Creative Solutions. XENIX, MS TM MICROSOFT INC., CP/M TM DRI, AMIGA, C64, C128 TM COMMODORE Int.

CIRCLE NO. 208 ON READER SERVICE CARD

that gives instructions for installing and operating the assembler and the other programs and a reference manual that defines the syntax of assembly language directives and operand expressions. The instruction set, however, is not defined. A list of instruction mnemonics is provided in an appendix, but no explanation is given of the functions and operands of each one. The descriptions of the pseudo-operations (the directives that control the operation of the assembler, as opposed to instructions that are executed at runtime) are clear and complete, but the explanation of address operands leaves much to be desired. Terms such as *relocatable operand* and *absolute symbol* are used without being defined. This manual is meant for a reader already skilled in assembly language in general and the 8088 family in particular.

IBM's documentation looks quite impressive in comparison. It consists of two volumes, each about the size of the DOS manual: one is a user's guide, the other a reference manual. Adequate white space and three-color printing make for a very slick visual presentation. The first sentence in the user's guide warns the reader that the manuals and, in fact, the assembler itself are meant for experienced assembly language programmers. However, the overall effect of both the appearance and the content is less daunting for the newcomer than is Microsoft's effort. The major portion of IBM's reference manual is devoted to the machine instruction set. For each instruction, it gives an explanation of its operation (including execution logic), the various acceptable operands, the machine language format, and the flags affected. The manual does not claim to be a tutorial, so it has no explanation of why or where to use each instruction, only of how. The descriptions of writing address expressions are much more informative than Microsoft's, and information on the pseudo-ops is at least as good. The 8087 numeric processor is given a very complete description. The structure of cross-reference files produced for input to the CREF utility is documented, allowing the extension of this utility to other languages (BASIC, for example).

The IBM documentation has several flaws. The two volumes are poorly organized. The reference manual contains the descriptions of the processor instructions and the pseudo-operations, but the syntax of the assembly language (for example, use of operators, such as OFFSET and BYTE PTR) is spelled out in an early chapter of the user's guide. Thus, the user needs to refer to both volumes to write the source program. A more logical method might be to group all information pertaining to the source program in one volume and explanations about assembling, linking, and testing in the other. Alternatively, one volume could be devoted to the information specific to this assembler, the other to the general information on the instruction set and hardware-related subjects. The IBM manuals also have some omissions; for example, the location counter reference ($) is implemented but not mentioned.

The biggest disappointment, however, is that this very impressive-looking pair of manuals just misses being a complete reference set for the assembly language programmer. The assembler-specific information is quite good, and the hardware-related information on the instructions, memory segmentation, addressing modes and the 8087 is excellent as far as it goes. An additional two or three chapters on the architecture, registers, and interrupt structure of the 8088/80286 processors would provide everything the programmer needs. As it is, the user needs yet a third volume that fills in the hardware details. A list of suggested titles is given; they are all reference books, not tutorials for beginners at assembly language programming. Such a reference will duplicate much of the information in IBM's documentation, but will not replace either one of the volumes because of the above-mentioned splitting of assembler-specific topics between the two.

The documentation from both Microsoft and IBM fails to explain the assembler's strong typing. Because this is an unusual feature for an assembler, even experienced programmers may be unfamiliar with it. Neither assembler's documentation defines in one section all the types assigned by the assembler and the legal uses of each; instead, the user must infer the typing rules from the many that are given for forming assembler expressions. Microsoft's documentation does a slightly better job in this regard. It, at least, specifies the type checking that is performed for symbols that refer to data in memory.

Both the IBM and Microsoft assemblers provide a significant increase in programming power over previous versions. Overall, their capabilities are quite impressive, especially in conditional assembly and defining macros. They compare favorably with the features of many mainframe assemblers.

Current users of MASM will certainly want to upgrade to either an IBM or a Microsoft assembler, but which one? Microsoft must be the choice if the full potential of the 286 chip is to be realized; this capability puts it a version ahead of the IBM product. The IBM documentation is so much better that it too deserves consideration, especially for use on 8088-based machines. If a choice must be made, the nod goes to Microsoft 3.01, especially for users already familiar with MASM, because of its added functionality on the PC/AT and because of the MAKE utility.

Ted Mirecki is a corporate planner responsible for developing decision support systems on a variety of hardware. He has a master's degree in computer science.

LISTING 1: LST1ASM.BAS

```
100 ' PROGRAM TO CONVERT .LST FILE CREATED BY MASM TO .ASM FILE
110 '     NOTE:  SET OBJLEN TO 40 IF LISTING WAS PRODUCED
120 '            WITH .CRF FILE
130 OBJLEN = 32
140 '
150 CLS
160 PRINT "Enter name of .LST file (no extension)"
170 INPUT; FIL$
180 OPEN FIL$+".LST" FOR INPUT AS #1
190 OPEN FIL$+".ASM" FOR OUTPUT AS #2
200 PRINT " "
210 ROW = CSRLIN
220 COUNT = 0
230 PRINT COUNT; "     records written"
240 '
250 '-------------------------- MAIN PROCESS LOOP
260 IF EOF(1) GOTO 610
270 LINE INPUT #1, IN$          'READ A LINE FROM LST FILE
280 IF LEN(IN$)=0 GOTO 260      'IGNORE NULL LINES
290 '
300 ' HANDLE NEW PAGE:  IF START OF SYMBOL TABLE, THEN ALL DONE
310 '                   OTHERWISE, SKIP 4 LINES OF HEADING
320 IF LEFT$(IN$,1) <> CHR$(12) GOTO 420
330   IF INSTR(38, IN$, "Symbol") GOTO 610
340   FOR X = 2 TO 5
350     IF EOF(1) GOTO 610
360     LINE INPUT #1, IN$
370   NEXT
380   GOTO 320
390 '
400 ' IGNORE LINES GENERATED BY MACRO.
410 '     THEY HAVE + IN COL OBJLEN-1, NOT PRECEDED BY TABS
```

```
420 IF MID$(IN$,OBJLEN-1,1) = "+" GOTO 260
430 '
440 'SCAN INPUT LINE, COUNTING CHARACTERS AND EXPANDING TABS
450 '  UNTIL OBJLEN CHARACTERS ARE COUNTED OR END OF INPUT LINE.
460 '  IF END OF LINE, IGNORE IT; OTHERWISE WRITE OUT REMAINDER
470 '
480 C = 0 : X = 1
490 FOR X = 1 TO LEN(IN$)
500    IF MID$(IN$,X,1)=CHR$(9) THEN C=C+8-C MOD 8 ELSE C=C+1
510    IF C < OBJLEN GOTO 570
520    PRINT #2, MID$(IN$, X+1)
530    COUNT = COUNT + 1
540    LOCATE ROW,1
550    PRINT COUNT
560    GOTO 580
570 NEXT
580 GOTO 260          '------------- END OF MAIN LOOP
590 '
600 '--------------------------------- WRAPUP PROCESS
610 PRINT " "
620 CLOSE
630 PRINT "File "; FIL$; ".ASM ready"
640 END
```

LISTING 2: MACTEST.ASM

```
          PAGE
          PAGE    ,132
TITLE     MACTEST Macro expansion benchmark

; THIS PROGRAM IS NOT EXECUTABLE.
; USE ONLY FOR TIMING MACRO GENERATION.

.NEST  = 0
.IFNUM = 0
.MAXIF = 0

MIF       MACRO   ARG1,CONDITION,ARG2
.NEST  = .NEST  + 1
.MAXIF = .MAXIF + 1
.IFNUM = .MAXIF
          STARTIF %.IFNUM
          PUSH    AX
          MOV     AX,ARG1
          CMP     AX,ARG2
          POP     AX
          CONDITION %.IFNUM
          ENDM

STARTIF   MACRO   WHERE
          .IF&WHERE = 2
          ENDM

MELSE     MACRO
          MELSE1  %.IFNUM
          ENDM

MELSE1    MACRO   WHERE
          JMP     ??END&WHERE
          .IF&WHERE = 1
??ELSE&WHERE:
          ENDM

MENDIF    MACRO
          MENDIF1 %.IFNUM
          .NEST = .NEST - 1
     if .NEST
          .IFNUM = .IFNUM - 1
          .X = .IFNUM
          .Y = .IFNUM
          REPT    .X
            MENDIF2 %.IFNUM
            ife .Y
              EXITM
            endif
          ENDM
     endif
          ENDM
MENDIF1   MACRO WHERE
     if .IF&WHERE - 1
??ELSE&WHERE:
     endif
??END&WHERE:
          .IF&WHERE = 0
          ENDM

MENDIF2   MACRO   WHERE
     if .IF&WHERE
          .Y = 0
     else
          .IFNUM = .IFNUM - 1
     endif
          ENDM

; Note that each condition generates a jump on the reverse
; condition.  That's because for the expression IF EQ, a jump
; to the ELSE part is taken if the result is unequal.

EQ        MACRO   WHERE
          JNE     ??ELSE&WHERE
          ENDM

NE        MACRO   WHERE
          JE      ??ELSE&WHERE
          ENDM

GT        MACRO   WHERE
          JLE     ??ELSE&WHERE
          ENDM

GE        MACRO   WHERE
          JL      ??ELSE&WHERE
          ENDM

LT        MACRO   WHERE
          JGE     ??ELSE&WHERE
          ENDM

LE        MACRO   WHERE
          JG      ??ELSE&WHERE
          ENDM
CSEG      SEGMENT
          ASSUME  CS:CSEG,DS:CSEG
TEST      PROC
          JMP     CODE1
A         DW      ?
B         DW      ?
C         DW      ?
D         DW      ?
E         DW      ?
F         DW      ?
; For 10 iterations, change to IRP X,<1,2,3,4,5,6,7,8,9,0>

          IRP     X,<1>
CODE&X:   MIF     A,EQ,B
            MOV   WORD PTR C,1234
            MOV   AX,A
            MOV   BX,B
            ADD   AX,BX
            MIF   B,GT,C
              SUB AX,C
            MELSE
              NEG AX
              ADD AX,C
            MENDIF
          MELSE
            MIF   D,LE,F
              ADD BX,E
            MELSE
              MIF F,NE,A
                SUB BX,AX
                ADD AX,E
              MENDIF
            MENDIF
          MENDIF
          ENDM

TEST      ENDP
CSEG      ENDS

STACK     SEGMENT STACK
          DW      32 DUP(?)
STACK     ENDS
          END     TEST
```

What to do out of

When you suddenly run out of memory, a creative use of the tools at hand can be deeply satisfying but very expensive.

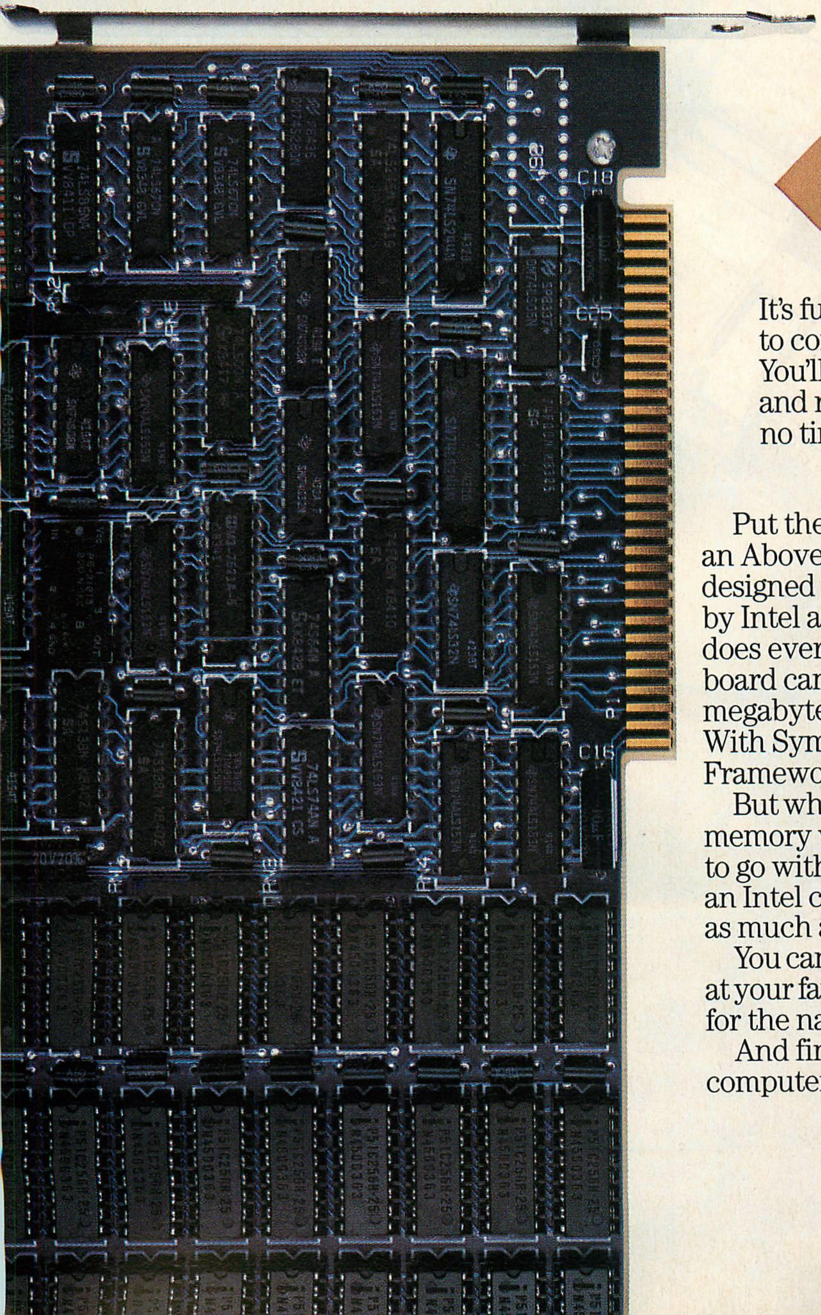
Keeping your database on two files...

Don't jump! If you survive, you'll have to deal with insurance forms as well as your memory problem.

is no problem.

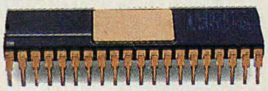
when you run memory.

Above™ Board can take you as far as four megabytes beyond 640K when you're working with Symphony,™ 1-2-3,® or Framework.™



It's fun to convert. You'll be up and running in no time!

Put the hammer away. Close the window. Pick up an Above Board. It's a new kind of memory board designed to meet the specification jointly developed by Intel and Lotus Development Corporation. It does everything any other self-respecting memory board can do. But it can also take you up to four megabytes beyond 640K where you need it most: With Symphony release 1.1, 1-2-3 release 2,* and Framework.**



But what good is all that memory without the speed to go with it? Intel has an answer for that, too. With an Intel coprocessor, your IBM PC® or PC AT™ can run as much as five times faster. Or more.

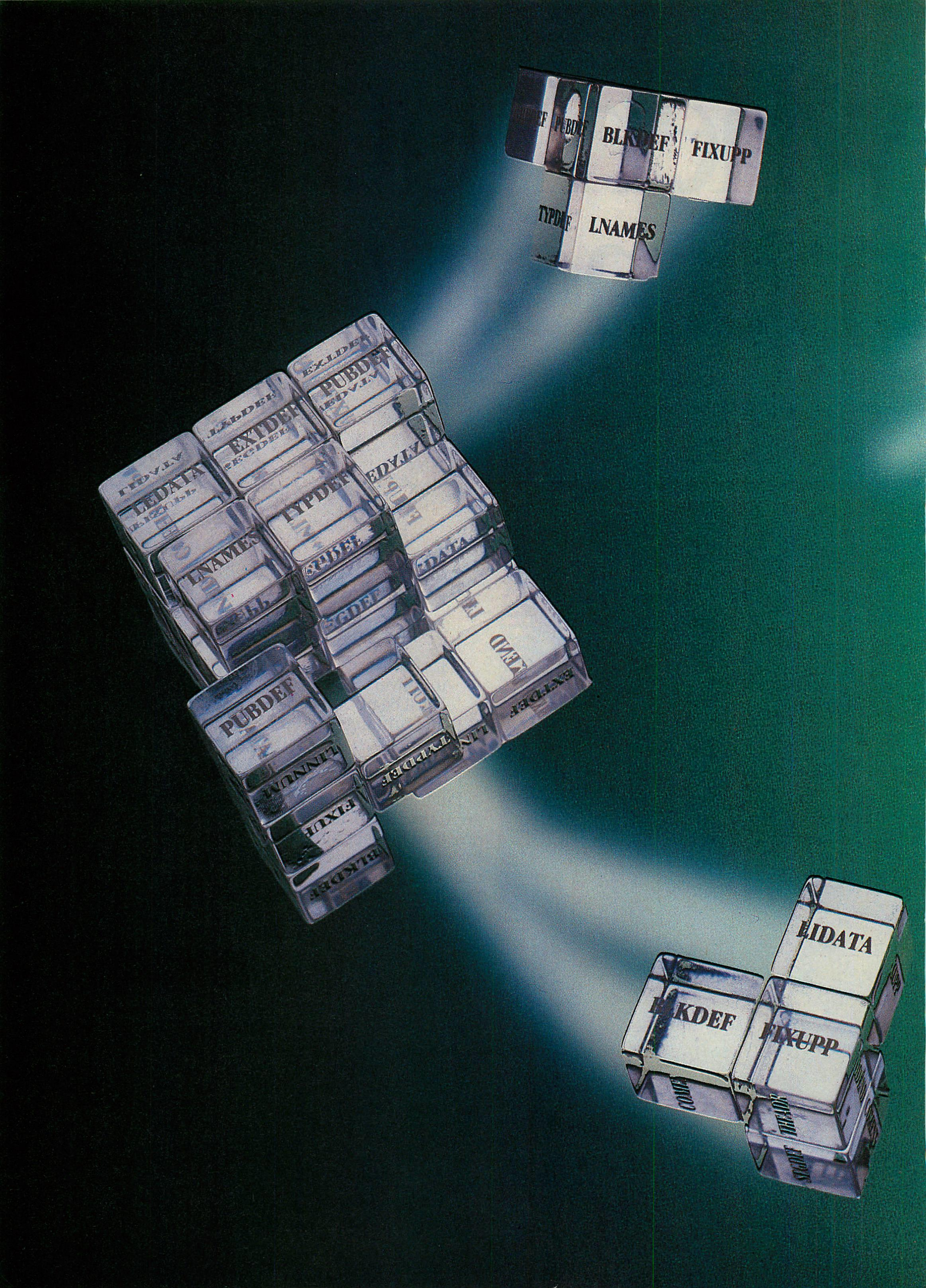
You can see and touch and buy these Intel products at your favorite computer store. Or call 800-538-3373 for the names of dealers near you.

And find out how to turn your average personal computer into an overachiever.

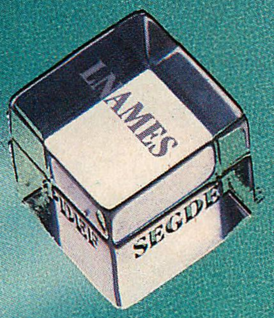
Help for the overachiever.

intel®

CIRCLE NO. 216 ON READER SERVICE CARD



BLKDEF
FIXUPP
TYPDEF
LNAMES
PUBDEF
EXTDEF
LEDATA
TYPDEF
LNAMES
PUBDEF
LIDATA
FIXUPP



.OBJ Lessons

The format of object modules is the key to combining code generated by different compilers, assemblers, and linkers.

STEVEN ARMBRUST and TED FORGERON

The ability to combine modules produced by different compilers and assemblers is important for effective program development. Such a capability offers the benefits of both high-level languages (faster and more error-free code, easier maintenance) and assembly language (for complex parts or for those parts in which extra speed is required).

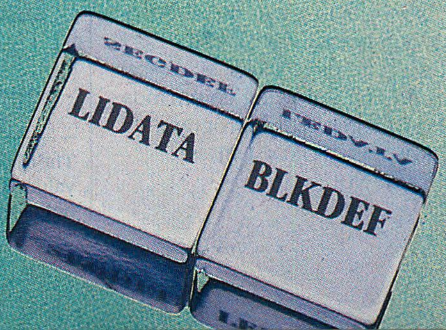
Combining a high-level language with assembly language is possible because the output of the different compilers and assemblers (called object code) is compatible. This low-level compatibility allows programmers to mix and match the languages. (Mixing two high-level languages calls for other considerations as well. Programmers must coordinate procedure calling conventions, stack usage, segment usage, and more. These considerations will not be discussed in this article.)

Compilers and assemblers translate source statements into instructions that the PC's microprocessor understands. The output of these compilers and assemblers (also called translators) consists of more than just machine instructions, however. It also contains information about main modules and subroutines, the ordering and grouping of segments of code, and source line numbers. This extra information is used by the linker and the loader to ensure that programs execute properly; it is also used by debuggers to help find and correct errors in the programs.

The reason most PC assemblers and compilers are compatible at the object level is that they all generate object code that adheres to the Microsoft standard—code that can be processed by the LINK program. This standard is a subset of a more powerful and comprehensive standard defined by Intel (the Intel standard describes code the Intel linker, LINK86, can process).

This article examines a typical Microsoft object module (with references to object code generated by the Microsoft Pascal compiler). This exami-

PHOTOGRAPH • MARC DAVID COHEN



nation illustrates exactly what kind of information each object module contains and points out areas in which the Microsoft standard deviates from the way Intel originally intended for object modules to appear.

THE OBJECT MODULE

The object module is the standard unit of information produced by a program translator (such as a compiler or assembler). The module is used as input to a program linker, which combines object modules and produces an executable file (such as an .EXE file on the IBM PC). The module can also be used by a program librarian to create a library of modules for selective linking.

When a library is linked with other object modules, not all of the modules in the library are used in the creation of the final file; the linker selects those referenced by the modules being linked. Figure 1 shows this transformation of source code into object modules and finally into an executable file.

Each object module consists of groups of data called *records*. The basic structure of a record is shown in figure 2. The Record Type field is a single byte indicating a record type. The Microsoft object module format has 15 different kinds of records. Refer to table 1 for a list of those records and a brief overview of each. The complete Intel standard defines 30 types of records. Table 2 contains a list of the additional records supported by Intel.

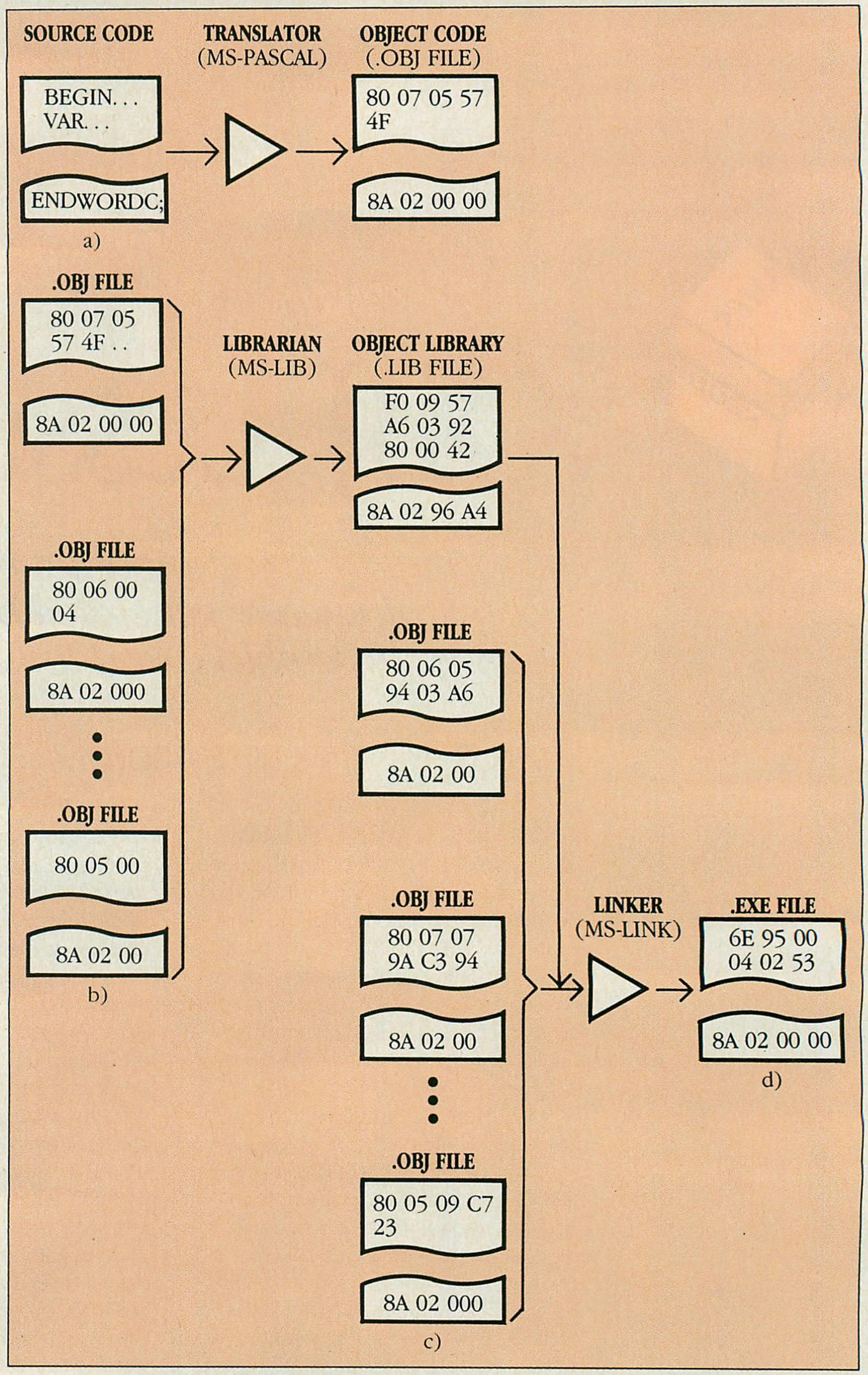
The Contents field is the meat of the record. For example, in data records, this portion can contain the actual machine code and an indication of the segment in which that code resides. The Record Length field is a two-byte value (a word) that specifies the number of bytes in the variable-length Contents field. The Checksum field helps the linker or loader identify data errors when reading the module from disk.

Every record in the object module has this basic format. The Contents field varies widely among record types, but the Record Type field identifies the type of record and how the Contents field is used. Figure 3 lists the formats of all Microsoft object module records.

Listing 1, called WORDC, is a simple Pascal program that counts the number of words in a text file and displays that number on the screen. WORDC was compiled using MS-Pascal version 3.20. The following commands were used to compile the program:

```
PAS1 WORDC.PAS, WORDC.OBJ,
   WORDC.LST, NUL;
PAS2
```

FIGURE 1: *Linking Process*



A translator, such as MS-Pascal, transforms the source code into object code (a). The object code can be added to a library of object modules (b), or it can be used as input to a linker (c). The output from the linker is an executable file (d).

FIGURE 2: *Record Structure*

RECORD TYPE	RECORD LENGTH	CONTENTS	CHECKSUM

The Contents field is the meat of the record. The Record Length field is a two-byte value that specifies the number of bytes in the Contents field. The Checksum field helps the linker or loader identify data errors when reading the module from disk.

The following command links an object code to a Pascal runtime library and produces an executable file:

```
LINK WORDC.OBJ, WORDC.EXE,
   WORDC.MAP, PASCAL.LIB;
```

The object module that results from the compilation of this program (WORDC.OBJ) is shown in figure 4. The left portion of the listing is a hexadecimal dump; the right portion is the corresponding ASCII translation. The start of each record and the record type are noted in the listing.

For clarity, this article refers to each record type by a six-character record name, rather than by a record type number. Intel and Microsoft also use these names to refer to the records in their documentation.

THEADR record (offset 0H). The first record in the object file is a translator header record. It lists the name of the object module without a file extension (in this case, WORDC).

The THEADR record in figure 4 starts at the first byte of the listing (offset 0H). This byte, and any others in the listing, can be found by using the numbers in the OFFSET column (the leftmost column). Each OFFSET entry lists the hexadecimal byte offset of the first byte in that row.

The format of a THEADR record is shown in figure 3-a. The Length and Checksum fields (which every record contains) in this example are structured as described earlier. In the THEADR record in figure 4, the two-byte value listed in the Length field is 07H 00H. This value might be misinterpreted to be 0700H, an awfully large header record. However, word values on Intel-based computers are interpreted with the low-order byte first, followed by the high-order byte. Thus, the length of this record is seven bytes, which defines the length of the Module Name field (a type of Contents field).

The Module Name field in the THEADR record is variable in length; the first byte defines the number of bytes remaining in the Module Name field (a number between 0 and 40), and this number is followed by hexadecimal representations of ASCII characters.

In figure 4, the Module Name field starts with the value 05, indicating that the next five bytes make up the module name. The ASCII characters represented in those bytes are *WORDC*, which is the name of the sample program.

COMENT record (offset 0AH). The next two records are COMENT records, which contain comments that do not affect program execution. COMENT records are created by program translators for use with other utility programs or for personal reference. COMENT records are used more by Intel than by Microsoft development tools. Refer to figure 3-b for the format of a COMENT record.

When the No Purge bit is set to 1, object file utility programs (such as linkers and librarians) should never delete this COMENT record. When the bit is set to 0, however, the utility programs may delete the record. When the No List bit is set to 1, it indicates that utility programs that normally list COMENT records should not list the text of this particular record.

TABLE 1: *Microsoft Object Record Types*

NUMBER (HEX)	TYPE	DESCRIPTION
7A	BLKDEF	Indicates the start of a program block, listing its location and return information.
7C	BLKEND	Indicates the end of a program block.
80	THEADR	Identifies start of modules generated by a translator.
88	COMENT	Denotes a nonexecutable comment.
8A	MODEND	Denotes the end of a module.
8C	EXTDEF	Lists the external variables to which a program refers.
8E	TYPDEF	Describes a variable type.
90	PUBDEF	Defines a program's public symbols.
94	LINNUM	Associates source code line numbers with corresponding object code.
96	LNAMES	Lists names of segments, classes, overlays, and groups.
98	SEGDEF	Describes a program segment.
9A	GRPDEF	Defines the segments that make up a group.
9C	FIXUPP	Identifies places in the data records at which the loader must change references to locations.
A0	LEDATA	Lists logical, enumerated data.
A2	LIDATA	Lists logical, iterated data.

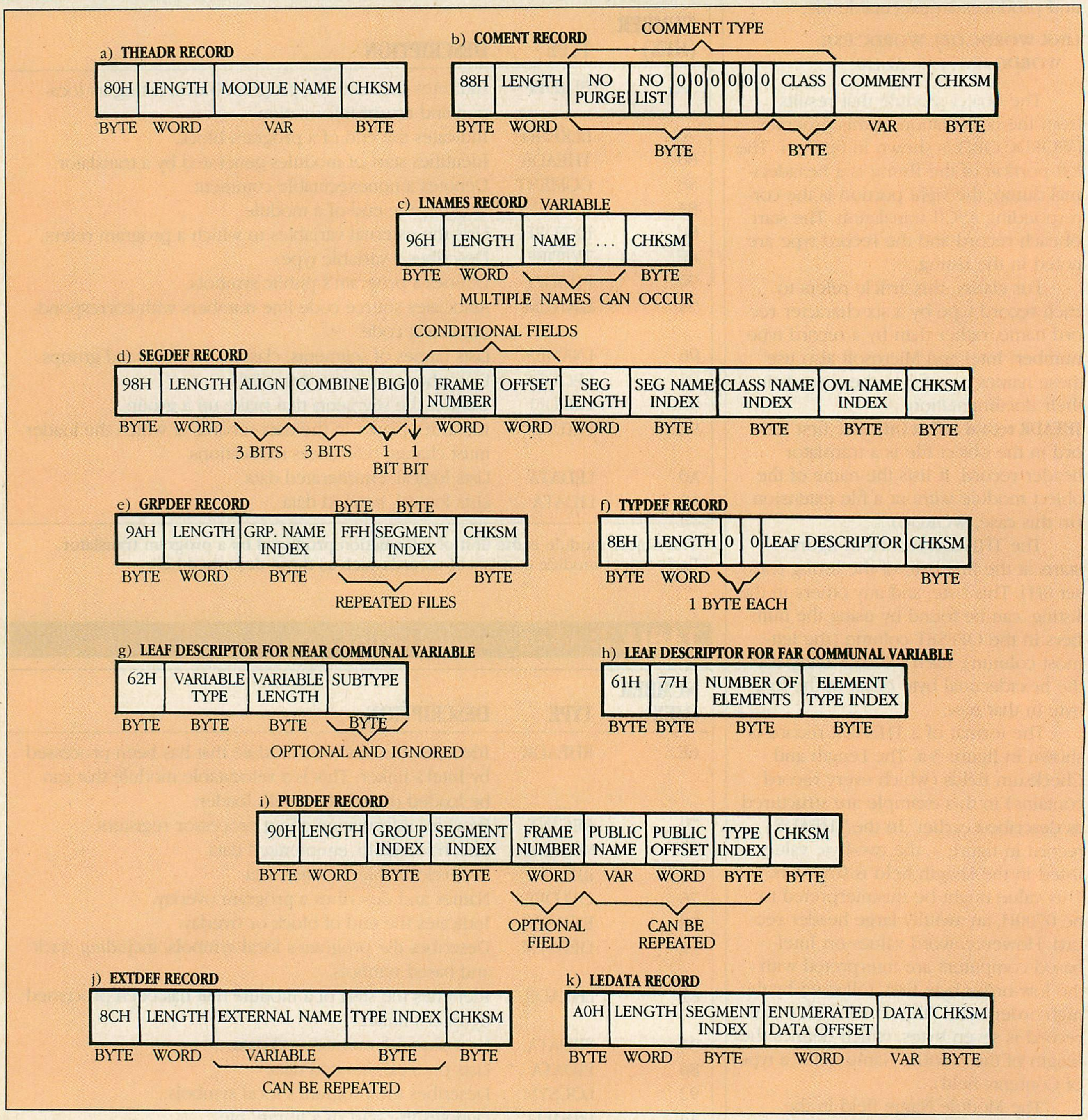
The object module is the unit of information produced by a program translator. Each object module consists of records such as those described here.

TABLE 2: *Additional Intel Record Types*

NUMBER (HEX)	TYPE	DESCRIPTION
6E	RHEADR	Identifies the start of a module that has been processed by Intel's linker. This is a relocatable module that can be loaded directly by Intel's loader.
70	REGINT	Provides information about processor registers.
72	REDATA	Lists relocatable, enumerated data.
74	RIDATA	Lists relocatable, iterated data.
76	OVLDEF	Names and describes a program overlay.
78	ENDREC	Indicates the end of block or overlay.
7E	DEBSYM	Describes the program's local symbols, including stack and based symbols.
82	LHEADR	Identifies the start of a module that has been processed by Intel's linker.
84	PEDATA	Lists physical, enumerated data.
86	PIDATA	Lists physical, iterated data.
92	LOCSYM	Describes the program's local symbols.
A4	LIBHED	Denotes the start of a library file.
A6	LIBNAM	Lists the names of the modules in a library file.
A8	LIBLOC	Lists the locations of all the modules in the library.
AA	LIBDIC	Names all the public symbols in the library.

The complete Intel standard defines 30 different types of records—15 more than Microsoft's standard uses. These additional record types are described here.

FIGURE 3: *Formats of Microsoft Object Records*

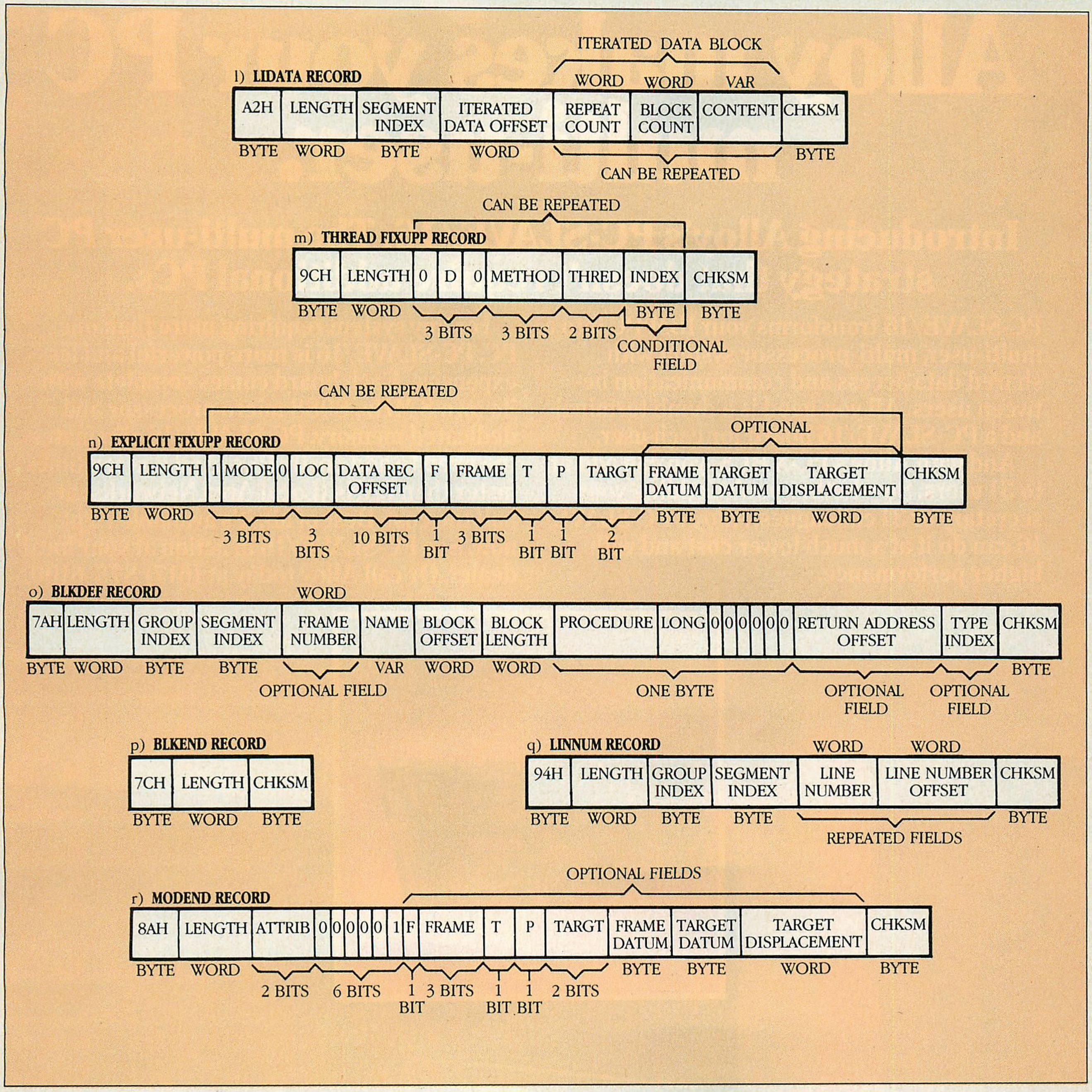


The Class field tells which kind of comment this record contains; this varies depending on which translator generated the object code. The field is not important, except that Intel normally reserves values 2-155 for its internal use. Notice that Microsoft uses a restricted value here (81H, or 129 decimal). This probably corresponds to a value that Intel uses in similar circumstances.

The variable-length Comment field contains the text that this record was created to represent. The two comment records in this object module start at bytes 0AH and 19H, respectively. They are used to identify the Microsoft Pascal compiler, as indicated by the text MS PASCAL and PASCAL in the respective Comment fields.

LNAMES record (offset 25H). LNAMES records contain a list of segments, classes, groups, and overlays that are used for reference by other record types. A *segment* is an area of memory that can be accessed with a 16-bit address, at most 64KB long. The starting address of a segment is referenced through one of the segment registers (CS, DS, ES, or SS). A *class* is several segments that are referenced by a common name (the class name). A *group* is one or more segments that fit into a single 64KB area of memory, can be referenced by a common name (the group name), and share the same segment register value. *Overlays* are separate sections of code that share a given memory area in order to optimize program size. Overlays sharing the same memory area cannot execute simultaneously.

Instead of duplicating the complete names of segments, classes, groups, and overlays in record after record, other object module records refer to these names by a number that indicates the



The formats of all the Microsoft object records are shown here. The complete Intel standard offers another 15 record types.

order in which the names appear in the LNAMES records. Number 1 is the first name in the first LNAMES record; the last number is the last name in the last LNAMES record. The format of an LNAMES record is shown in figure 3-c.

Each Name field consists of a one-byte length followed by the characters that specify the name of a segment, class, group, or overlay. As many name fields as necessary are allowed, with each one having the same structure.

The LNAMES record contained in figure 4 starts at byte 25H. It has a length of 44H bytes. The first name in the record is HIMEM and is five bytes long. The other names are LARGE, CONST, COMADS, DATA, STACK, MEMORY, HEAP, CODE, WORDC, a null name (as indicated by a length of 0), and DGROUP. Of all these names, only WORDC (the name of the program) is defined by the program shown in listing 1; the others are internally generated by the Microsoft Pascal compiler.

SEGDEF record (offset 6CH). SEGDEF (segment definition) records describe the segments that make up the program. Figure 3-d shows the format of a SEGDEF record. The first field after the standard Length field is Align, a three-bit field that specifies the alignment of the segment. The possible values in this field are described in table 3. Another three-bit field, the Combine field, describes the rules the linker must follow in order to combine this segment with others of the same name. This field has the values shown in table 4.

The Big field is a one-bit field that, if set to 1, indicates that the segment is exactly 64KB (65,536 bytes) long (note that a segment cannot be greater than 64KB in length). If the segment is less than 64KB long, this field is 0.

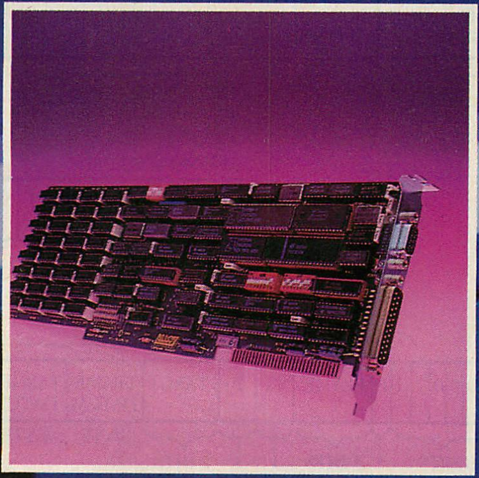
Alloy makes your PC multi-user.

Introducing Alloy's PC-SLAVE/16. The multi-user PC strategy that doesn't require additional PCs.

PC-SLAVE/16 transforms your IBM PC into a multi-user, multi-processor system with shared data access and communication between users. All you need is a dumb terminal and a PC-SLAVE/16 board for each user.

And with Alloy's PC-X BUS, you can add up to 31 users. So whether you're adding your secretary or 31 colleagues, Alloy's PC-SLAVE/16 can do it at a much lower cost and with higher performance.

For any system requiring more than one PC, PC-SLAVE/16 is more powerful and less expensive. Each board comes complete with its own 8 MHz, 8088 micro processor and dedicated memory. PC-SLAVE/16 eliminates the need to add costly PCs. And it is ten times faster than LANs (Local Area Networks) at half the cost. Best of all, it's available right now. See PC-SLAVE/16 in action at your local dealer. Or call Alloy today at (617) 875-6100.



ALLOY

ALLOY
Computer Products, Inc.

Alloy Computer Products, Inc., 100 Pennsylvania Avenue, Framingham, Massachusetts 01701. (617) 875-6100, TWX: 710-346-0394
In Europe: Alloy Computer Products (Europe) Ltd., Cirencester, Gloucestershire, England. Tel: 0285-68709, Tlx: 43340

CIRCLE NO. 112 ON READER SERVICE CARD

The Frame Num and Offset fields are present only for absolute segments (Align field contains 0). They identify the starting address of the absolute segment. The Frame Num field specifies the number of the frame containing the segment. A *frame* is a type of segment, using Intel terminology. It is a 64KB area of address space that begins on an even, 16-byte boundary (Intel defines this as a paragraph boundary). The frame beginning at address 0 is frame 0, the one beginning at address 10H (16 decimal) is frame 1, etc. The Offset is the number of bytes from the start of the frame at which the segment begins.

The Seg Length field is a word that lists the length of the segment in bytes. Because the largest value that a word can contain is 65,535, if the segment is exactly 64KB (65,535 bytes) long, this field is set to 0 and the Big field (described earlier) is set to 1.

The Seg Name Index, Class Name Index, and Ovl Name Index fields are single-byte fields that identify the segment's name, class, and overlay by specifying the numbers of the appropriate names in the LNAMES record. For example, a 2 indicates the second name in the LNAMES record, a 3 indicates the third name in the record, etc.

An examination of the first SEGDEF record in figure 4 will help to explain how SEGDEF records work. This record starts at byte 6CH. The first byte in the record (98H) identifies the record as a SEGDEF, while the next two bytes, 07H 00H, indicate the length of the rest of the record (7 bytes).

The next byte, 40H, represents the Align, Combine, and Big fields. To pick out the individual fields, 40H must be translated into its binary equivalent: 010 000 00. The leftmost three bits, binary 010 (or 2, in decimal form), give the value for the Align field. This indicates that the segment is a relocatable, word-aligned segment.

The rest of the bits are 0; therefore, the segment is a private segment (Combine field is 000) less than 64KB long (Big field is 0). Because the segment is not an absolute segment, the Frame and Offset fields are absent. The next word, EFH 00H, gives the length of the segment (239 decimal bytes).

The next three bytes (0AH, 09H, and 0BH) indicate the names of the segment, the class, and the overlay, respectively. These values are indices into the list of names in the LNAMES record. To interpret the names, go back to the LNAMES record (it starts at byte 25H in figure 4) and count to the tenth, ninth, and eleventh names. Counting to the tenth name reveals the segment name to be WORDC. The ninth name (the class name) is CODE, and the eleventh name, the overlay name, is null (this program has no overlays).

Examining the rest of the SEGDEF records (there are seven of them in all) shows that the program contains segments named WORDC, HEAP, STACK, DATA, COMADS, CONST, and HIMEM.

GRPDEF record (offset B2H). The record that follows the SEGDEF records is a GRPDEF (group definition) record. This record defines a group by naming it and by identifying all the segments that make up the group. (A group consists of several segments that fit into one 64KB area of memory. When these segments are combined as a group, one segment register can access any one of the individual segments.)

Figure 3-e illustrates the format of a GRPDEF record. In this record, the Grp Name Index, much like the other indexes that were described previously in this article is an index that specifies which of the names in the LNAMES record are used to identify the group.

The Segment Index identifies a segment that helps make up the group. It is a byte whose value is an index into the list of SEGDEF records discussed earlier. For example, a 1 indicates the first SEGDEF record, a 2 indicates the second SEGDEF record, and so on. As many Segment Indexes as necessary are allowed and each is preceded by the hexadecimal value FFH.

The GRPDEF record in figure 4 starts at byte B2H. It is 12 bytes long (0CH 00H), and its name is DGROUP (the twelfth name in the LNAMES list). This group has five segments, referencing the sixth, fifth, fourth, third, and second SEGDEF records in the module (HEAP, STACK, DATA, COMADS, and

TABLE 3: *Possible Align Values*

VALUE	MEANING
0	Absolute segment. In this case, both the Frame and Offset fields are present to describe the location of the segment.
1	Relocatable and byte-aligned segment (the segment can begin at any memory address).
2	Relocatable and word-aligned segment (the segment must begin at an address on a word boundary).
3	Relocatable and paragraph-aligned segment (the segment must begin at an address on a 16-byte boundary).
4	Relocatable and page-aligned segment (the segment must begin at an address on a 256-byte boundary).

The Align field is a three-bit field that specifies the alignment of the segment. The possible values in this field are listed and described here.

TABLE 4: *Possible Combine Values*

VALUE	MEANING
0	A private segment that cannot be combined, even with other segments of the same name, and that has its own unique base address.
2	A public segment. All data in segments of the same name and class are loaded into sequential memory locations as one segment and can be referenced with a single segment-register value.
5	A stack segment. All data in segments of the same name and class are loaded into sequential memory locations as one segment and can be referenced with a single segment-register value. The LINK program places the segment address into the SS register (the SP register contains the Offset field value if provided; otherwise, it contains no determined value) when an .EXE file is loaded for execution.
6	A common segment. All segments of the same name and class are loaded overlapped into a single segment (each module's segment reference begins at a memory address with offset 0). The size of the common segment is the size of the largest segment definition.

The Combine field, which can have the values listed above, describes the rules the linker must follow to combine the Align segment with others of the same name.

CONST). (This is the normal DGROUP produced by the MS-Pascal compiler. MS-Pascal loads the segment address for DGROUP into the DS register during the program's initialization.)

TYPDEF record (offset C1H). The next record is a type definition record. In the complete Intel format, TYPDEF records contain a wealth of information about public variables, external variables, code blocks, debugging symbols, and local symbols. These records do not affect program execution, but they do provide useful information to high-level language debuggers.

Microsoft translators, however, usually generate just a single dummy TYPDEF that is used as a placeholder reference for fields in other object module records. (Some other records need to reference a TYPDEF, and those records can always refer back to this dummy TYPDEF.) In cases in which a null TYPEDEF is used, almost all TYPDEFs can be ignored.

The one time TYPDEF records do contain meaningful information is when communal variables are involved. A *communal variable* is an uninitialized public variable whose size is not fixed at compilation time (such as a FORTRAN common block). With communal variables, the same variable might be defined with different size declarations in several different modules. (The only PC language that uses communal variables is FORTRAN.)

TYPDEF records appear for communal variables, describing the type and size of each. The linker then allocates space for the largest such variable defined, and all references to a given communal variable have an identical starting address and varying lengths.

The format of a TYPDEF record is shown in figure 3-f. In this record, the Leaf Descriptor has one of two formats, depending on whether the communal variable is a *near variable* (its address is described only with a 16-bit offset value) or a *far variable* (its 32-bit address contains both segment and offset values). For near variables, the first byte of the Leaf Descriptor is 62H (see figure 3-g). The Variable Type field is a byte that describes the type of the communal variable with the following values:

77H Array
79H Structure
7BH Scalar

The Variable Length field lists the length of the variable in bits. The Subtype, in the full Intel format, then specifies additional information about the variable. In the Microsoft format, this field has no meaning and, therefore, usually is not present.

If the communal variable is a far variable (figure 3-h), the first byte of the Leaf Descriptor field has a value of 61H. The next byte, 77H, means the variable is an array (see the list of variable types above). The *MS-DOS Programmer's Reference* states that arrays are the only far variables to appear.

The Number of Elements field lists the number of entries in the array. The Element Type Index is an index into the list of previously defined TYPDEF records. For example, a 2 would indicate the second TYPDEF record; the TYPDEF pointed to is a TYPDEF for a near variable, and it identifies the type of the array elements.

Examining the TYPDEF record shown in figure 4 (beginning at byte C1H) illustrates that this TYPDEF is simply a dummy record and can be ignored. The length (05H 00H) is five bytes. The next two bytes (00H 00H) are the 0 bytes shown in the TYPDEF format. The next byte (7BH) indicates that this variable is a scalar. The 0 byte that follows indicates that the length of the scalar is 0. Therefore, this TYPDEF record does not provide any real information; it merely defines a variable with length 0.

In Microsoft object records, the Type Index field refers to the dummy TYPDEF, because Microsoft languages do not generate TYPDEF records.

PUBDEF record (offset C9H). Next comes a PUBDEF (public names definition) record, which describes all the public symbols in this object module. *Public symbols* are names of variables, procedures, and functions that are defined in this module and that can be referenced by other modules. PUBDEF records provide information about these symbols in order that the linker can match them with external symbols that appear in other modules.

PUBDEF records list the names of public symbols, the group and segment in which they reside, and their offset from the start of the segment. Figure 3-i shows the format of the PUBDEF record. The Group Index is an index into the LNAMES record that identifies the name of the group associated with this public symbol. A value of 0 indicates that the symbol has no associated group. The Segment Index identifies the segment containing this symbol by referencing a SEGDEF record (a 1 indicates the first SEGDEF, a 2 indicates the second, and so on).

If both the Group Index and the Segment Index are 0, the Frame Number field is present; otherwise, the Frame Number field does not appear. If present, the Frame Number lists the number of a frame containing the public symbol. (Remember, a frame is a 64KB block of memory that starts on a 16-byte boundary. Frame 0 goes from 0 to 64K; frame 1 goes from 10H to 64K+10H; and so on.)

The Public Name, Public Offset, and Type Index fields are repeated for each public name defined in this record. The Public Name field lists the name of the public symbol; the first byte lists the number of bytes in the name, followed by an ASCII representation of the name. The Public Offset field (a word value) specifies the number of bytes (the offest) from the start of the segment or frame containing the symbol.

The Type Index field identifies the TYPDEF record that describes this symbol. Like the Segment Index field this field lists the number of a previously defined TYPDEF record (a 1 indicates the first TYPDEF, a 2 indicates the second, etc.). In Microsoft object records, this field refers back to the dummy TYPDEF, because Microsoft languages do not generate TYPDEF records.

The sample program shown in listing 1 did not explicitly define any public symbols, but the compiler generated two of them. An examination of the PUBDEF record in figure 4 follows, beginning at byte C9H.

The record length (16H 00H) is 22 bytes. The Group Index is 00H (no group is associated with these symbols). The Segment Index is 01H, indicating that the symbols are contained in the first segment that is defined in a SEGDEF record (WORDC).

The name of the first symbol is next. It is five bytes long and is called WORDC (the program name). Its offset is 1 (01H 00H), and its Type Index is also 1 (there is only one TYPDEF).

The next public symbol is six bytes long and has the name ENTGQQ. It also has an offset of 1 as well as a Type Index of 1. This means that the compiler has defined two different names for the same symbol. ENTGQQ is used by the Pascal runtime library as a common method for identifying the beginning of

the main program. (Any symbols that end with the letters *QQ* most likely have been generated by MS-Pascal.)

EXTDEF record (offset E2H). The EXTDEF record, which appears next in figure 4, defines the names of external symbols to which this program refers. *External symbols* are the names of variables or subroutines that a program uses but does not define (they are defined in other object modules and declared public there). Figure 3-j illustrates the format of an EXTDEF record.

The sample program declared no external subroutines or variables, but the compiler generated external references to some of its library routines. The EXTDEF record beginning at byte E2H of figure 4 lists them.

The first word (71H 00H) indicates the length of the record (113 bytes). The remaining portion of the record names the externally defined variables and identifies their TYPDEF records. This record lists symbols named BEGXQQ, WTLFQQ, WTIFQQ, WTSFQQ, RTCFQQ,EOFFQQ, PPEFQQ, RTAFQQ, PPMFQQ, NEWFQQ, INIFQQ, INPFQQ, OUTFQQ, and RESFQQ, all referencing TYPEDEF record 1 (which indicates a null record).

These external variables are defined in the Pascal runtime library. Because the compiler always generates its own external references, programmers must always link their Pascal programs to the Pascal libraries that accompany the MS-Pascal compiler.

LEDATA and LIDATA records (offset 164H). The next record in figure 4 is a FIXUPP record, which identifies locations in the data that must be changed before a program can be loaded into memory. To understand FIXUPP records, the user must know how data records work.

Data records contain the actual machine code and data that get loaded into memory. Two kinds of data records can appear in a Microsoft object module: LEDATA (logical enumerated data) and LIDATA (logical iterated data). An LEDATA record contains data just as they will appear when loaded into memory (except where they will be modified by FIXUPP records). The LIDATA record is encoded to compress repeated data, thereby making the object module smaller.

The format of an LEDATA record is shown in figure 3-k. In this record, both the Segment Index and Enumerated Data Offset fields help identify where the data will reside in memory. The Segment Index is the number of a previously defined SEGDEF record. The Enumerated Data Offset is a word that indicates where the data begin, relative to the start of that segment. The variable-length DATA field gets loaded into memory at the address specified by the Segment Index and Data Offset.

The first LEDATA record in listing 2 starts at byte 164H. This record is eight bytes long (08H 00H). The Segment Index (04H) identifies the fourth SEGDEF (DATA) as the segment containing this data. The address offset from the start of that segment is 0 (00H 00H). The remaining bytes of the record are the data that get loaded into the DATA segment. Other LEDATA records in this module can be interpreted in the same way.

Several LEDATA records are listed for this sample program. The largest of the records contains the program's code, which will reside in the code segment. Each LEDATA record can contain only 1,024 bytes of information; thus, if the sample program's code were longer, additional LEDATA records would be present. Other LEDATA records designate information for other segments, including program data such as the text *word count =*, which is placed into the CONST segment. (Listing 1 shows that these are words that the program displays on the screen.)

Data blocks can be nested as many as 17 levels deep, but the size of the iterated data block field cannot exceed 512 bytes.

The term *logical enumerated data* implies two characteristics of the data in the record. First, the data are logical. In Intel terminology, that means relocatable; the loader is free to load the data wherever necessary, based on available system memory and the software that is already executing.

Second, the data are enumerated, which means that every byte is listed, even if many consecutive bytes contain the same value. Another type of data (iterated) is encoded to decrease the size of the data record. Iterated data records compress consecutive bytes that contain the same value. Although the sample program contains no iterated data records, programs that load a lot of repeated data might.

The Segment Index and Iterated Data Offset fields contained in the LIDATA record are exactly the same as the similarly named fields in the LEDATA record (refer to figure 3-l); they identify an address for loading data. The next three fields in LIDATA (Repeat Count, Block Count, and Content) are called an *iterated data block* and specify the iterated data. Repeat Count is a word that indicates the number of times the Content field is to be repeated. If the Repeat Count is 3, three copies of the Content field will be loaded into memory.

The next field, Block Count, is a word that identifies whether the Content field contains only data or whether the Content field itself is made up of iterated data blocks (Repeat Count, Block Count, and Content fields). If the Block Count field is 0, the Content field contains only data, which are loaded into memory as many times as indicated by the Repeat Count field. However, if the Block Count field is not 0, the Content field itself contains iterated data blocks; that is, the iterated data blocks are nested. The number in the Block Count field specifies the number of iterated data blocks in the Content field.

If the Content field contains only data, the first byte indicates the number of bytes of data in the rest of the field. However, if the Content field does not contain data, it is interpreted as another iterated data block, with the first word being a Repeat Count.

Figure 5 illustrates how the iterated data block can be nested to compress the repeated data. The first part of the figure shows a simple iterated data block with a single Repeat Count and Block Count. The second part of the figure illustrates an iterated data block with three levels of nesting. This nesting of data blocks can go as many as 17 levels deep, as long as the size of the iterated data block field does not exceed 512 bytes (this is a limitation of the Microsoft linker).

If a program loads a great deal of repeated data (such as defining a large array initialized to a common value), LIDATA records allow that repeated data be stored in very few bytes. Of course, LIDATA records are not appropriate when no repeated data are present. Compilers and assemblers are free to decide which record to use; they may determine which will cause the resulting object file to be smaller.

FIXUPP record (offset 156H). FIXUPP records identify data in either LEDATA or LIDATA records that refer to symbols whose locations will change as a result of the linker deciding how to relocate everything. FIXUPP records are notes to

FIGURE 4: *Object Module from WORDC.OBJ*

```
Hexidecimal Dump of File: b:wordc.obj

OFFSET                    HEX                            ASCII

        \/-THEADR                     \/-COMENT
0000    80 07 00 05 57 4F 52 44 43 F5 88 0C 00 00 00 4D    ....WORDC......M

                                   \/-COMENT
0010    53 20 50 41 53 43 41 4C F8 88 09 00 00 81 50 41    S PASCAL......PA

                       \/-LNAMES
0020    53 43 41 4C 3A 96 44 00 05 48 49 4D 45 4D 05 4C    SCAL:.D..HIMEM.L
0030    41 52 47 45 05 43 4F 4E 53 54 06 43 4F 4D 41 44    ARGE.CONST.COMAD
0040    53 04 44 41 54 41 05 53 54 41 43 4B 06 4D 45 4D    S.DATA.STACK.MEM
0050    4F 52 59 04 48 45 41 50 04 43 4F 44 45 05 57 4F    ORY.HEAP.CODE.WO

                                            \/-SEGDEF
0060    52 44 43 00 06 44 47 52 4F 55 50 E4 98 07 00 40    RDC..DGROUP....@

                          \/-SEGDEF
0070    EF 00 0A 09 0B 14 98 07 00 44 00 00 08 07 0B 03    .........D......

        \/-SEGDEF                     \/-SEGDEF
0080    98 07 00 54 00 00 06 06 0B F6 98 07 00 48 12 05    ...T.........H..

                    \/-SEGDEF                     \/-SEGDEF
0090    05 05 0B ED 98 07 00 48 00 00 04 04 0B 06 98 07    .......H........

                                \/-SEGDEF
00A0    00 48 1C 00 03 03 0B EC 98 07 00 48 00 00 01 01    .H.........H....

              \/-GRPDEF
00B0    0B 0C 9A 0C 00 0C FF 06 FF 05 FF 04 FF 03 FF 02    ................

           \/-TYPDEF               \/-PUBDEF
00C0    3F 8E 05 00 00 00 7B 00 F2 90 16 00 00 01 05 57    ?.....{........W
00D0    4F 52 44 43 01 00 01 06 45 4E 54 47 51 51 01 00    ORDC....ENTGQQ..

              \/-EXTDEF
00E0    01 FB 8C 71 00 06 42 45 47 58 51 51 01 06 57 54    ...q..BEGXQQ..WT
00F0    4C 46 51 51 01 06 57 54 49 46 51 51 01 06 57 54    LFQQ..WTIFQQ..WT
0100    53 46 51 51 01 06 52 54 43 46 51 51 01 06 45 4F    SFQQ..RTCFQQ..EO
0110    46 46 51 51 01 06 50 50 45 46 51 51 01 06 52 54    FFQQ..PPEFQQ..RT
0120    41 46 51 51 01 06 50 50 4D 46 51 51 01 06 4E 45    AFQQ..PPMFQQ..NE
0130    57 46 51 51 01 06 49 4E 49 46 51 51 01 06 49 4E    WFQQ..INIFQQ..IN
0140    50 46 51 51 01 06 4F 55 54 46 51 51 01 06 52 45    PFQQ..OUTFQQ..RE

                          \/-FIXUPP
0150    53 46 51 51 01 19 9C 0B 00 00 06 01 04 02 02 03    SFQQ............

                    \/-LEDATA                \/-LEDATA
0160    01 44 01 01 A0 08 00 04 00 00 42 57 02 00 B9 A0    .D........BW....
0170    10 00 04 05 00 57 4F 52 44 43 2E 50 41 53 20 20    .....WORDC.PAS

              \/-LEDATA               \/-LEDATA
0180    20 56 A0 05 00 04 04 00 0C 47 A0 10 00 06 02 00     V.......G......

                                            \/-LEDATA
0190    07 49 4E 5F 46 49 4C 45 4C 41 4C 00 52 A0 3F 00    .IN_FILELAL.R.?.
01A0    01 01 00 55 8B EC 81 EC 04 00 9A 00 00 00 00 B8    ...U............
01B0    90 02 50 B8 50 00 50 B8 01 00 50 9A 00 00 00 00    ..P.P.P...P.....
01C0    B8 14 00 50 B8 50 00 50 B8 01 00 50 9A 00 00 00    ...P.P.P...P....
                                                     \/-FIXUPP
01D0    00 B8 07 00 50 B8 03 00 50 9A 00 00 00 00 E8 9C    ....P...P.......
01E0    1A 00 CC 08 56 0B C4 0D 8D CC 19 56 0A C4 1E 8D    ....V......V....

                                            \/-LEDATA
01F0    CC 2A 56 0A C4 33 8C CC 37 56 09 C8 A0 12 00 06    .*V..3..7V......
0200    0E 00 0D 77 6F 72 64 20 63 6F 75 6E 74 20 3D 20    ...word count =

           \/-LEDATA
0210    AB A0 B7 00 01 3C 00 B8 00 00 50 B8 14 00 50 9A    .....<....P...P.
0220    00 00 00 00 9A 00 00 00 00 B8 14 00 50 9A 00 00    ............P...
0230    00 00 C7 06 0C 05 00 00 C6 06 10 05 00 B8 14 00    ................
0240    50 9A 00 00 00 00 D1 E8 72 4A B8 14 00 50 B8 0E    P.......rJ...P..
0250    05 1E 50 33 C0 50 B8 FF 00 50 9A 00 00 00 00 80    ..P3.P...P......
0260    3E 0E 05 20 74 15 80 3E 0E 05 09 74 0E 80 3E 0E    >.. t..>...t..>.
0270    05 0D 74 07 80 3E 0E 05 0A 75 07 C6 06 10 05 00    ..t..>...u......
0280    EB 10 F6 06 10 05 01 75 09 C6 06 10 05 01 FF 06    .......u........
0290    0C 05 EB A9 B8 00 00 50 B8 0D 00 50 B8 0F 00 1E    .......P...P....
02A0    50 B8 FF 7F 50 50 9A 00 00 00 00 B8 00 00 50 FF    P...PP........P.
02B0    36 0C 05 B8 FF 7F 50 50 9A 00 00 00 00 B8 00 00    6.....PP........

                                         \/-FIXUPP
02C0    50 9A 00 00 00 00 8B E5 5D CB 03 9C 64 00 C4 01    P.......]...d...
02D0    86 0C C4 05 8D CC 09 56 08 CC 0E 56 07 C4 13 8D    .......V...V....
02E0    CC 17 56 0E C4 1D 8D C4 23 8D C4 27 8D CC 2B 56    ..V.....#..'..+V
02F0    06 C4 34 8D C4 38 8D CC 44 56 05 C4 4A 8D C4 51    ..4..8..DV..J..Q
0300    8D C4 58 8D C4 5F 8D C4 66 8D C4 6D 8D C4 74 8D    ..X.._..f..m..t.
0310    C4 79 8D C4 7E 86 0D C4 86 8C CC 90 56 04 C4 95    .y..~.......V...
0320    86 0D C4 9A 8D CC A2 56 03 C4 A7 86 0D CC AB 56    .......V.......V

              \/-BLKDEF
0330    02 B2 7A 11 00 00 01 05 57 4F 52 44 43 01 00 EE    ..z.....WORDC...

                          \/-BLKEND   \/-LINNUM
0340    00 60 00 00 00 A1 7C 01 00 83 94 3B 00 00 01 11    .'....|....;....
0350    00 4E 00 12 00 57 00 13 00 5D 00 14 00 62 00 16    .N...W...]...b..
0360    00 6F 00 17 00 84 00 1B 00 A0 00 1C 00 A7 00 1E    .o..............
0370    00 AE 00 1F 00 B3 00 20 00 B7 00 21 00 B7 00 22    ....... ...!..."

                                \/-MODEND
0380    00 B9 00 23 00 EB 00 AE 8A 02 00 00 74             ...#........t
```

The first byte of each record comprising WORDC.OBJ is marked with a *V* symbol, with the record type written beside it.

the linker that essentially say, "Here is a place (called a *location*) where the code refers to a symbol (called a *target*). If the target is moved around, the location's reference in the code should be changed accordingly."

FIXUPP records supply several pieces of information that the linker needs to fix up a location's reference. Figure 6 illustrates such information. First, the FIXUPP record identifies the location to be fixed up. It specifies this as an offset into the data portion of the data record (a FIXUPP always applies to the closest previous LEDATA or LIDATA record). Next, the FIXUPP record identifies the target—that is, the symbol being referenced by the location.

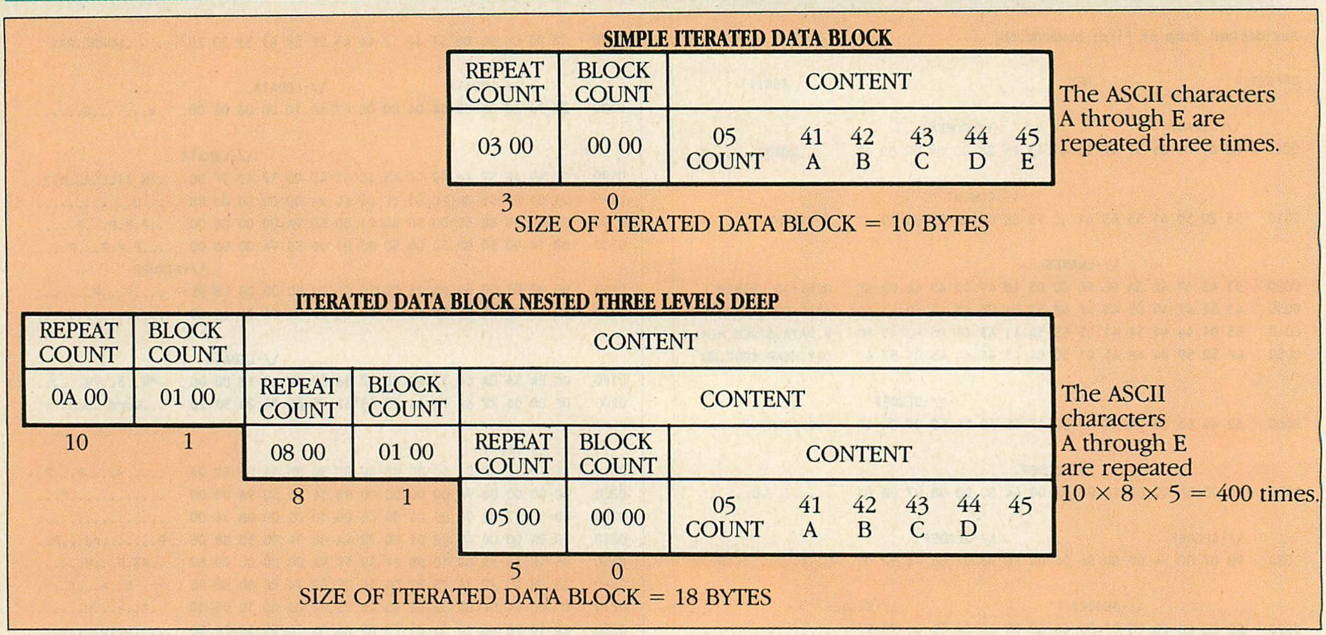
Once the target and location are identified, the FIXUPP record provides information about the framework of the fix-up, that is, how to change the location so that it accurately refers to the target. There are two such pieces of information: the *mode* and the *frame*.

The mode tells the linker whether to change the location's reference to the target using a value relative to the location itself (self-relative mode) or to make the change in the target reference relative to the start of some segment (segment-relative mode). In self-relative mode, the location and target reside in the same segment, and the linker is required to change only the offset portion of the location's reference to the target. In segment-relative mode, the linker needs to change both the base and the offset reference to the target.

The frame is a 64KB area of memory starting on a 16-byte boundary. It provides the information the linker needs to change the base portion of the target's address. For self-relative references, this frame is usually the segment referenced in the address of the location and target. For segment-relative references, the frame is the segment referenced in the address of the target.

Knowing the mode and the frame, the linker is able to change any location's reference accurately, whether or not the target resides in the same segment as the location.

FIGURE 5: *Iterated Data Block*



The iterated data block can be nested to compress repeated data. In the LIDATA record, this block can be nested 17 levels deep.

FIXUPP records are of two types: *thread* and *explicit fix-up*. A thread FIXUPP record defines locations in memory and the symbols to which they refer. Explicit FIXUPP records reference threads in the same manner as SEGDEF records establish the relationship between a location, target, and frame. By defining commonly used information only once, thread records allow explicit FIXUPP records to be shorter. The format of thread and explicit FIXUPP records is shown in figures 3-m and 3-n.

In a thread FIXUPP, the D field is a single bit that indicates the kind of thread in question. If D is 0, the thread is a target thread, and the information supplied here is used to identify the target of the fix-up (the symbol to which the location in memory refers). If D is 1, the thread is a frame thread, which supplies information about the frame (a segment, a 64KB area of memory starting on a 16-byte boundary). The linker needs to know the frame in order that it can correctly adjust the location's reference to the target.

The Method field is a three-bit field that lists the method the linker must use to identify the target or the frame, based on the value of the D field. If D=0 (a target thread), Method can have one of the values shown in table 5. If D=1 (a frame thread), Method can have one of the values shown in table 6.

The Thread field is a two-bit field that assigns a number (from 0 to 3) to the thread being defined. Later, FIXUPP records can refer to targets or frames using these numbers, instead of having to define the target or frame explicitly. A single thread FIXUPP record can define up to four target threads and four frame threads. If a later thread FIXUPP record assigns the same number to a new thread, it redefines that thread number for remaining FIXUPP records.

The Index field identifies the SEGDEF, GRPDEF, or EXTDEF index referred to by the Method field. For example, if this is a target thread that uses method T0 (see table 5), the Index field is an index into the list of SEGDEF records (a 1 indicates the first SEGDEF defined, a 2 indicates the second SEGDEF, and so on). Likewise, if this thread were a frame thread that used method F2, the Index field would be an index into the EXTDEF list of external symbols.

The Index field always appears except for frame threads that use methods F3, F4, or F5. In those cases, the Index field is unnecessary. The combination of D, Method, Thread, and Index fields defines a single thread. Each thread FIXUPP can define four target threads and four frame threads.

An examination of one of the threads in the first FIXUPP record in figure 4 (starting at byte 156H) will help explain how thread FIXUPPs work. The length of the record (0BH 00H) is 11 decimal bytes. The next byte (00H) contains the D, Method, and Thread fields for the first thread. Because the byte is all 0s, the hexadecimal value does not need to be expanded into binary. It is easy to see that D=0 (this is a target thread), Method is T0 (the target is specified by a SEGDEF index and an offset), and the thread number is 0.

The next byte (06H) is an index that identifies this target. Because the Method field indicates a SEGDEF index, the 06H value is an index into the list of SEGDEF records. It identifies the sixth SEGDEF defined (the CONST segment). When later FIXUPP records refer to target thread 0, they will be referring to the CONST segment.

The other four threads in this record can be interpreted in the same way. Target thread 1 refers to segment 4 (the DATA segment). Target thread 2 refers to segment 2 (the HEAP segment). Target thread 3 refers to segment 1 (the WORDC segment). Frame thread 0 refers to group 1 (DGROUP).

A thread FIXUPP record merely provides common information to be used by other explicit FIXUPP records. An explicit FIXUPP record identifies the location that is to be fixed up, the symbol (or target) to which the location refers, and the context (or frame) in which this fix-up is to take place. Using this information, the linker is then able to adjust the addresses.

In the explicit FIXUPP record in figure 3-n, the Mode field is a single bit that indicates whether the fix-up is self-relative (mode 0) or segment-relative (mode 1). A self-relative FIXUPP lets the linker know that it needs to support 8- and 16-bit offsets without segment values (because the program uses only call, jump, and short jump instructions).

Segment-relative FIXUPPs, however, require, the linker to support all 8088 addressing modes.

The Loc field is a three-bit field that identifies the kind of location to be fixed up. The possible values for this field are shown in table 7.

The Data Rec Offset field is a 10-bit field that identifies the start of the location to be fixed up. It is the offset from the start of the data portion of the preceding data record (either LEDATA or LIDATA). For example, if the Loc field identified the location as a *lobyte* and the Data Rec Offset were 2, the FIXUPP record would tell the loader to change the address of the second byte in the previous data record. Instructions for changing this byte are supplied later in the FIXUPP record.

The F field in the FIXUPP record is a single bit that indicates how the frame for this fix-up is specified. The frame is the context in which the fix-up occurs; that is, it idenifies a 64KB block of memory address space, starting on a 16-byte boundary, that will eventually contain the target as soon as the data are loaded into memory. During system operation, the current frame depends on the contents of a segment register. Therefore, the frame provides a starting point (or base) that the linker can use in order to generate an accurate address when it performs the fix-up. If F=0, the FIXUPP record specifies the frame explicitly. If F=1, the FIXUPP record refers to the previous thread FIXUPP for the location of the frame.

The Frame field is a three-bit field whose meaning depends on the setting of the F field. If F=0 (an explicit frame), the Frame field contains a number from 0 to 5 that lists the method of identifying the frame. These methods of identifying frames (F0 through F5) are the same as those discussed earlier with the thread FIXUPP record.

If F=1 (reference to a previous thread), the Frame field identifies the number of a previously defined frame thread. For example, if the Frame field is 1, then the frame of this fix-up is frame thread number 1, as defined in the previous thread FIXUPP record.

The T field is a single bit that tells whether the target of this fix-up (the symbol to which the location refers) is specified explicitly (T=0) or in a previous thread FIXUPP (T=1).

The P field is a single bit that tells whether the target is specified in a primary way (P=0) or a secondary way (P=1). Primary ways correspond to methods T0 through T3 outlined earlier. They require both an index (to identify the segment, group, or external symbol) and an offset. Secondary ways correspond to methods T4 through T7 and do not require an offset (or the presence of the Target Displacement field in the FIXUPP record).

The Targt field is a two-bit field whose meaning depends on the value of the T field. If T=0 (explicit specification), Targt indicates the method of identifying the target. The possible values, 0 through 3, correspond to methods T0 through T3 if P = 0 and T4 through T7 if P = 1. These methods were described earlier in the discussion of thread FIXUPPs. If T=1 (reference to a previous thread), the Targt field identifies a previously defined target thread. In this case, the number is the same number of a target thread defined in the previous thread FIXUPP.

The Frame Datum field appears only when the frame is specified explicitly (F=1). It is an index into the list of SEGDEFs, GRPDEFs, or EXTDEFs (which list is used depends on the method identified in the Frame field).

The Target Datum field is similar to the Frame Datum field. It appears when the target is specified explicitly

TABLE 5: *Possible Method Values (D=0)*

VALUE	MEANING
0	The target is identified by a SEGDEF index and an offset. (T0)
1	The target is identified by a GRPDEF index and an offset. (T1)
2	The target is identified by an EXTDEF index and an offset. (T2)
3	The target is identified by a frame number and an offset. (T3)
4	The target is identified by a SEGDEF index only. The target starts at the beginning of the segment. (T4)
5	The target is identified by a GRPDEF index only. (T5)
6	The target is identified by an EXTDEF index only. (T6)
7	The target is identified by a frame number only. (T7)

The Method field lists the method the linker must use to identify the target or frame based on the value of the D field. IF D=0, Method can have any value shown here.

TABLE 6: *Possible Method Values (D=1)*

VALUE	MEANING
0	The frame is identified by a SEGDEF index. The frame is the highest-numbered frame that contains the segment. (F0)
1	The frame is identified by a GRPDEF index. (F1)
2	The frame is identified by an EXTDEF index. (F2)
3	The frame is identified by a frame number. (F3)
4	The frame is the highest-numbered frame that contains the segment of the location to be fixed up. The loader can determine this segment because the LEDATA record (which contains the location) indicates into which segment the data will be loaded. (F4)
5	The frame is the same as that of the target. (F5)

If the value of the D field is 1, indicating a frame thread, the Method field can have any one of the values shown in this list.

TABLE 7: *Possible Loc Values*

VALUE	MEANING
0	Lobyte (the low-order byte of a word).
1	Offset (the low-order word of a pointer).
2	Base (the high-order word of a pointer).
3	Pointer.
4	Hibyte (the high-order byte of a word).

The Loc field is a three-bit field that identifies the kind of location to be fixed up by the FIXUPP record. The possible values for this field are shown here.

Technical Bulletin
No. 2 in a series.

SUBJECT: Engineering a LAN for Maximum Flexibility.

Quantum Software Systems Ltd. proudly announces QNX 2.0 — the Ultimate Distributed Network Operating System. QNX 2.0 is now available for the IBM-PC, IBM-AT, PC compatibles, DEC Rainbow and TANDY 2000. If you have been waiting for a Real-time Multi-tasking Multi-user Operating system with fourth generation LAN support, then QNX 2.0 can offer you today what the competition can't even begin to promise for the future.

QNX 2.0 integrates the Local Area Network architecture right into the heart of the operating system, at the fundamental level of intertask communication allowing tasks to communicate transparently with other tasks across the whole network. This means that any task (program/application) may access ANY serial port, ANY printer or ANY disk on the network. There are no artificial restricitons. Every PC with a disk is a potential file server. PCs without disks will automatically BOOT over the network.

QNX on the IBM-PC AT:

QNX is the first Multi-tasking Multi-user Operating system available for the AT. It is available in both networked and single machine configurations. At about 2.5 times faster than the QNX 8088 PC based systems, and 10 times faster than other multi-tasking operating systems on the same processor, QNX is the ideal program development environment.

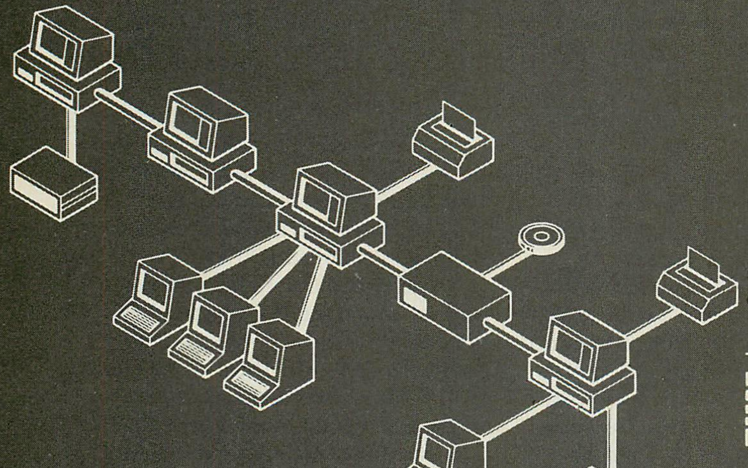
O/S	Computer	Processor	Measured time
QNX ™	IBM-PC AT	80286	480 usec
XENIX ™	Intel-286	80286	4,930 usec

File Security:

Designed with extensive file security features, QNX 2.0 provides login protection with network wide file permission checking based on 255 groups of 255 users. In addition, each PC user may control network access to devices attached locally to their machine.

Distributed Processing:

The QNX LAN supports distributed processing as well as distributed devices. Tasks may be executed on remote stations as easily as they may be executed on the local work station. This allows pure processing elements (PCs without keyboards or displays) to be plugged into the network to be used as an un-committed processing resource. This is ideal for real-time, process control, data acquisition and data communication applications.



Global Communications:

QNX supports a full implementation of X.25 allowing connection to public networks such as Telenet and Datapac. This allows you to link geographically separate LANs together providing true global area networking.

Cost Effective Growth and Flexible Solutions:

QNX is affordable, and will work with the PCs you use today and those you will use tomorrow. You may mix and match different brand PCs on the same QNX network with absolute ease. Multi-user expansion may be accomplished by adding terminals to PCs or PCs to the network. You can start your multi-user application on a single PC with 1 to 10 attached terminals. Once your single processor starts to show signs of degradation, add another PC and connect terminals to the new processor. If the disk becomes the major bottleneck, you may add hard disks to other attached PCs to distribute the processing. Applications which are very CPU intensive may wish to limit a single user to each processor and expand the system with low cost diskless PCs used as work stations. QNX does offer a truly cost effective and flexible solution to your applications needs.

Portability:

QNX 2.0 is portable. The operating system is independant of the physical local area network. It is available in a form suitable for porting to other 8088/8086/80186/80286 computers in the consumer, educational and industrial market place. QNX is ROMable and can operate in as little as 128Kb RAM.

DOS Compatibility:

PC-DOS version 2.1 can run as a task under the QNX 1.2 or 2.0 operating systems. QNX will also allow transparent access to the DOS file system partition and floppies.

QNX Products:

- QNX Operating System
- Full Screen Multi-terminal Editor
- Extended Utilities
- C Compiler & 8086 Assembler
- Basic Compiler
- Qbol (dibol) Compiler
- Text Processor
- Real Time Spelling Checker
- PC-DOS Emulator
- Electronic Mail
- Electronic Teleconferencing
- Full Screen Menu Developer
- Isam File Utility
- Networking Board
- OEM Customization Kit (to port QNX)

Established:

Quantum sold over 10,000 copies of its operating system during 1984, into all business systems environments, to developers of real time applications, government and educational systems, to software developers/integrators, universities and research establishments.

QUANTUM SOFTWARE SYSTEMS LTD.

Moodie Drive, HiTech Park, 215 Stafford Rd.
Ottawa, Canada K2H 9C1 (613) 726-1893

(T=1) and is an index into the list of SEGDEFs, GRPDEFs, or EXTDEFs.

Finally, the Target Displacement field is a two-byte field that appears only when P=0 (the primary method of specifying a target). In this case, the Target Displacement is an offset from the start of the SEGDEF, GRPDEF, or EXTDEF (whose index appeared either in the Target Datum field or in an earlier thread FIXUPP).

The first record in the sample program serves as an example of how an explicit FIXUPP record works. This record starts at byte 1DFH. Its length (1AH 00H) is 26 decimal bytes. The next two bytes (CCH 08H) specify the Mode, Loc, and Data Rec Offset fields. The binary format of the hexadecimal numbers is as follows:

1 1 0 011 0000001000

The leftmost bit indicates that this actually is an explicit FIXUPP record (a 0 would have indicated a thread FIXUPP). The next bit, the Mode bit, indicates that the mode is segment-relative.

The third bit is always set to 0, and the three bits that follow make up the Loc field. The value of these bits (011 binary or 3 decimal) indicates that the location to be fixed up is a pointer.

The next 10 bits specify the location to be fixed up. This value (8 decimal) means that the location starts at the eighth byte from the start of the data portion of the previous data record (an LEDATA record starting at byte 19DH in figure 4). The data portion of that LEDATA record starts at byte 1A1H in figure 4, and the eighth byte is byte 1A8H (whose value is 04). This byte then becomes the first byte of a pointer the compiler has indicated for change.

The next byte in the FIXUPP record (56H) contains the values for the F, Frame, T, P, and Targt fields. To interpret these fields, this byte must be translated into binary, as follows:

0 101 0 1 10

The first bit is the F bit. Because it is set to 0, it indicates that this record specifies the frame explicitly, rather than by referring to a thread FIXUPP. Therefore, the next field (101) indicates the method of specifying the frame. Because 101 binary translates into 5 decimal, the method F5 is used. F5 says the frame is determined by the target. So for this fix-up, the information that specifies the target will also specify the frame.

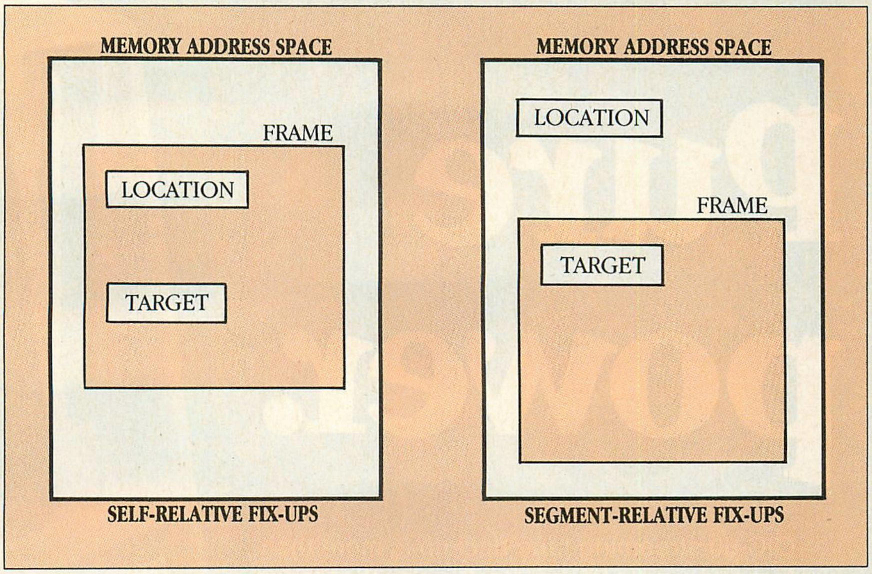
The T bit is next. Because it is set to 0, it indicates that the target will also be specified explicitly and not by referring to a thread FIXUPP. The bit that follows is the P bit. Because it is 1, the target specification is made in the secondary way. That is, the index (in the Target Datum field) alone specifies the target. No displacement is needed, and therefore no Target Displacement field appears in this fix-up.

The last two bits in this byte (10) indicate the method of specifying the target. This value (2 decimal) means that method T6, an index into the list of external symbols, is used.

The next byte (0BH), the last one in this fix-up, is the Target Datum field. Notice that the Frame Datum field does not appear in this fix-up because the frame is specified implicitly by the target. The Target Displacement field does not appear either, because the target is specified in a secondary way. The remaining bytes in the record are additional complete fix-ups. Thus, this Target Datum field is an index that identifies both the target and the frame. Because the method of specifying the target was listed as T2, this is an index into the list of external symbols described in the EXTDEF record, and it identifies symbol 0BH (or 11 decimal). In the EXTDEF record (beginning at byte E2 in figure 4), the eleventh symbol listed is INIFQQ.

If the P bit had been set to 1, an additional displacement field would have identified the displacement from INIFQQ to which the fix-up should refer. Because P was set to 0, the location to be fixed up (identified earlier) depends solely on the address of the external symbol INIFQQ.

FIGURE 6: *The FIXUPP Record*



FIXUPP records supply pieces of information the linker needs to fix up a location's reference. The relationship between these pieces of information is shown here.

Thus, to summarize the first fix-up in this FIXUPP record, the location to be fixed up is a pointer that begins eight bytes into the previous data record and that points to the external symbol INIFQQ. When the location of INIFQQ becomes known (by linking the program to another module that declares INIFQQ as a public symbol), the linker will change the pointer to refer directly to INIFQQ. The linker knows exactly how to change the reference because it also knows the frame (the segment register value) that is in effect when the target is loaded into memory.

This FIXUPP record contains six more fix-ups. The one just outlined did not refer to any of the threads defined earlier, but some of the later ones do. To see some of the variations in fix-ups, just continue this process and translate some of the others shown in this listing.

BLKDEF and BLKEND records (offsets 332H and 346H). After some more LEDATA and FIXUPP records, a BLKDEF (block definition) and a BLKEND (block end) record appear. Together, these two records describe a program block. A program block can define such language constructs as procedures, loops, and multiline if-then-else statements, depending on the language and compiler implementation. A BLKDEF and a BLKEND record exist for every procedure and for every program block that has its own local variables.

The BLKDEF record identifies the group and segment containing the block. It also lists the name of the block, the block's offset from the start

Pure power.

QIC-60 streaming tape backup is more than just fast and simple to operate. It's also the leader in powerful processing.

For example, our unique five-head tape deck offers immediate read-after-write verification. No rewinding necessary. And no re-recording of the entire backup after an error.

Another powerful advantage is automatic adjustment of the head mechanisms to seek the best track alignment every time a tape is inserted. Plus our exclusive power sensor is designed to alert and freeze the write head if PC power is reduced, which eliminates the chance of disk damage. We're so confident in the technical excellence of QIC-60 that we offer an aggressive one-year limited warranty on the entire line, compared to the 90-day warranty provided by other manufacturers.

Best of all are a pair of backup bonuses that turn QIC-60 into a sophisticated data management tool. When you select mirror-image restore, QIC-60 sees bad disk sectors and automatically reroutes data to safe locations for improved data reliability.

Tecmar offers AT owners a special enhancement with an internally-mounted QIC-60 tape drive that costs even less than external systems. And when you add our internal AT Hard Disk to the package, you have a fully upgraded system ready to use today.

Pure speed. Pure power. Pure innovation from Tecmar.

For a free demonstration of the new standard in tape backup, see your Tecmar dealer or call us at **216/349-1009** for the location of the dealer nearest you.

Tecmar
THE POWER BEHIND THE PC

6225 Cochran Road Solon, Ohio 44139

CIRCLE NO. 225 ON READER SERVICE CARD

of the segment, and the length of the block. In addition, if the BLKDEF record describes a procedure, the record provides information about the type of procedure. The format of a BLKDEF record is shown in figure 3-o.

The Group Index indexes the names in the LNAMES record. It identifies the group associated with the block. The Segment Index is an index into the list of SEGDEF records. It identifies the segment containing the block. If both these values are 0, a Frame Number field is present, indicating the number of the frame containing the block.

The Name field identifies the name of the block. As in all name fields, the first byte indicates the number of bytes in the rest of the name. The remaining bytes are ASCII values.

The Block Offset field is a word that indicates the block's offset from the start of the segment containing the block. The Block Length field is a word that lists the number of bytes in the block. The Procedure bit indicates whether the block is a procedure. If the bit is 1, the block is a procedure; however, if it is 0, the block is some different type (such as a DO loop).

The Long bit has meaning only if the Procedure bit is set to 1. When the Long bit is set to 1, it implies that the procedure's return address is a four-byte value (both CS and IP). When the bit is set to 0, the procedure has a two-byte return address (IP only).

The Return Address Offset field is present only if the block is a procedure. This field is a word that gives the location of the procedure's return address on the stack. The return address gets pushed onto the stack when the procedure is called. The Return Address Offset is interpreted as an offset from the BP register, which points to the return address on the stack.

As usual, the Type Index field identifies the number of the TYPDEF record that defines this block.

The BLKDEF record in figure 4 starts at byte 332H. The length (11H 00H) is 17 decimal bytes. The Group Index is 00H; thus, this block is not associated with a group. The Segment Index (01H) points back to the first SEGDEF defined, indicating this block is part of the segment WORDC.

The next field is the Name field. The first byte (05H) gives the number of bytes in the name. The next five (57H 4FH 52H 44H 43H) are ASCII codes for the block name (WORDC).

Next comes the Block Offset (01H 00H), indicating that the block starts one byte after the start of the segment.

TABLE 8: *Line Number/Line Number Offset Pairs*

SOURCE LINE NUMBER	OBJECT CODE OFFSET
11H (17 decimal)	4EH
12H (18 decimal)	57H
13H (19 decimal)	5DH
14H (20 decimal)	62H
16H (22 decimal)	6FH
17H (23 decimal)	84H
1BH (27 decimal)	A0H
1CH (28 decimal)	A7H
1EH (30 decimal)	AEH
1FH (31 decimal)	B3H
20H (32 decimal)	B7H
21H (33 decimal)	B7H
22H (34 decimal)	B9H
23H (35 decimal)	EBH

Each Line Number and Line Number Offset pair identifies the location of one line of source code. One such pair exists for each executable source line in the program.

TABLE 9: *Possible Attrib Values*

VALUE	MEANING
0	Non-main module with no starting address listed.
1	Non-main module with starting address listed.
2	Main module with no starting address listed.
3	Main module with starting address listed.

The Attrib field of the MODEND record is a two-bit field that specifies the attributes of the module. Possible values for this field are listed here.

The Block Length (EEH 00H) indicates that the block is 238 decimal bytes long. The next byte (60H) contains the settings for the Procedure and Long bits. In binary, this byte translates into 1100000. Therefore, this block is a procedure; appropriately, it does have a four-byte return address.

Because the block is a procedure, the next word (00H 00H) is the Return Address Offset. This value indicates that the procedure's return address is at BP+0. The byte that follows (00H) is the Type Index. The 0 value here indicates that no TYPDEF record is associated with this block.

A BLKEND record appears next, indicating the end of the block defined by the previous BLKDEF record. In the full Intel object module format, the combination of BLKDEF and BLKEND records provides information about the scope of variables. Between the BLKDEF and BLKEND records appear debugging symbol records that describe each symbol defined in the block. However, the Microsoft format does not currently support these debugging records; BLKEND records therefore serve no useful purpose.

The format of a BLKEND record is shown in figure 3-p and contains no special fields. The BLKEND record in figure 4 starts at byte 346H.

LINNUM record (offset 34AH). After the BLKEND record comes a line numbers record that lists the address of each executable line of source code. Although not necessary for program execution, it provides information debuggers can use to associate the source code with the translated object code.

The format of a LINNUM record is shown in figure 3-q. In this record, the Group Index field is an index into the LNAMES record and indicates the group containing this code. Likewise, the Segment Index is an index into the list of SEGDEF records, identifying the segment containing the code.

Each Line Number and Line Number Offset pair identifies the location of one line of source code. The Line Number field is a word that lists the source line number. The Line Number Offset field is a word that lists the address of that line, relative to the start of the segment. A Line Number/Line Number Offset pair exists for each executable source line in the program.

The LINNUM record of the sample program starts at byte 34AH of figure 4. Its length (3BH 00H) is 59 decimal bytes. Its Group Index is 00H (no group), and its Segment Index is 01H (the first SEGDEF, WORDC). Next come the Line Number/Line Number Offset pairs. These pairs are listed in table 8.

To see those lines included in this record, compile the source code in listing 1 and look at the file WORDC.LST.

MODEND record (offset 388H). The last record in the file is the module end record. It indicates the end of the module that was begun by THEADR, whether that module is a main module or a subprogram, and sometimes the module's starting address. Figure 3-r shows the record's format.

The Attrib field of the MODEND record is a two-bit field that specifies the attributes of the module. Possible values for this field are listed in table 9.

The rest of the fields (F, Frame, T, P, Targt, Frame Datum, Target Datum, and Target Displacement) specify the starting address. These fields appear only if the Attrib field indicates that the starting address is listed. The fields are interpreted in the same way as the equivalent fields in the FIXUPP record.

In figure 4, the MODEND record starts at byte 388H. The length of the record (02H 00H) is two bytes. The Attrib field is 0, which indicates a nonmain module with no starting address listed. The other fields are not present. If this sample program had included more than one module, another THEADR record would have appeared to indicate the start of the next module.

THE EXPANDED STANDARD

Even though Microsoft defines 15 different record types, these are just a subset of the record types available in the full Intel set (see table 2). What do these extra records provide, and why did Microsoft choose to omit them?

Some of the records (such as PEDATA, PIDATA, REDATA, RIDATA, RHEADR, LHEADR) indicate kinds of relocatable or absolute data that Microsoft deals with in other ways. For example, Intel's relocatable object modules (REDATA and RIDATA) correspond to Microsoft's .EXE files. Intel's absolute object modules (PEDATA and PIDATA) correspond to Microsoft's .COM files. The correspondence is not exact, but .EXE and .COM files provide enough features that Microsoft does not need to support those additional records.

Other object records, such as the full Intel implementation of TYPDEF, DEBSYM, or LOCSYM, are debugging records with information about the source program. Such information allows symbolic and high-level language debuggers to provide additional services to programmers, such as displaying local variables or producing formatted dumps of structures.

Two probable reasons Microsoft decided not to include these debugging records are code size and compiler development time. The object files generated by the Intel compilers contain more information, so they are larger. In the early days of the PC, when DOS supported only single-sided (160KB) floppy disks, it was more important to ensure that a program would fit on a disk than it was to provide elaborate debugging information. Therefore, all unnecessary records were eliminated. Although larger-capacity disks are now supported by the PC, most compilers and linkers have not yet been modified to add the extra records.

The other reason for the lack of these extra records might be the time necessary to write a compiler. Requiring a compiler to generate extra records means additional effort by the compiler writer. Further, because IBM DEBUG, the first debugger for the PC (and the only one for a long time) does not use this information, adding records probably seemed like wasted effort. Omitting the extra records got the compilers out to the public sooner with no noticeable loss of features.

Today's market, however, demands debugging tools that are more advanced than IBM DEBUG. High-level language debuggers, which allow programmers to debug while viewing source code (not an assembly language version of it), are possible, but they require that object modules contain more information than current PC compilers now generate. To lay the groundwork for these advanced debugging tools, Microsoft and other companies soon may add debugging records to the object modules generated by their compilers.

The format of the object module is the key to compatibility between different compilers, assemblers, linkers, and librarians. Products from many different companies use the same format (the one recognized by the LINK program). This standard format allows programmers to combine code generated by many different products.

Besides linking object modules, programmers can also combine them into libraries so that the individual modules can be pulled out selectively by the linker. These library files contain additional records that provide this selective linking capability.

REFERENCES

Intel Corporation. *8086 Relocatable Object Module Formats. An Intel Technical Specification.* (Santa Clara, CA: Intel Corp., 1981).

Microsoft Corporation. *MS-DOS Programmer's Reference Manual.* (Bellevue, WA: Microsoft Corp., 1984).

Steven Armbrust is a freelance technical writer. Ted Forgeron is a microcomputer software consultant. They work primarily in the "Silicon Forest" near Portland, Oregon.

LISTING 1: WORDC.PAS

```
{ WORDC -- Counts the number of words in a file }

PROGRAM wordc(in_file,output) ;

  CONST
    tab = 9 ;
    lf = 10 ;
    cr = 13 ;

  VAR
    in_file,out_file : text ;
    count : integer ;
    input_char : char ;
    inword : boolean ;

  BEGIN {wordc}
    reset(in_file) ;
    count := 0 ;
    inword := false ;
    WHILE NOT eof(in_file) DO
      BEGIN
        read(in_file,input_char) ;
        IF (input_char=' ') OR
          (input_char=chr(tab)) OR
          (input_char=chr(cr)) OR
          (input_char=chr(lf)) THEN
          inword := false
        ELSE IF ( NOT inword) THEN
          BEGIN
            inword := true ;
            count := count + 1 ;
          END ;
      END ; {while}
    writeln('word count = ',count) ;
  END. {wordc}
```

NEW STUFF!

IF YOU PROGRAM IN 'C' DON'T TURN THE PAGE!

PADLOCK

RECOMMENDED!

Protect Any Program Using Anyone's Diskettes

You are a program developer seeking a low-cost method to protect against piracy? You are with a prominent company fearful of embarrassing lawsuits should employees copy programs illegally?

We have a devilishly clever solution for the two of you we call PadLock™. It gives you the keys to padlock any program you want—yours or someone else's—without buying a stockpile of expensive fingerprinted disks.

PadLock is software. One of its programs formats any manufacturer's disk, embedding your secret code in sectors the operating system will ignore. Another program installs this code in any program files you name, whether YourProg or WordStar™, or any .COM or .EXE files. From then on there's no picking the lock without the combination: the coded software together with a coded disk.

It's not despised copy prevention. Protected programs can be copied anywhere: to hard disks, to backup diskettes. It's a lock and key scheme. A protected program will just lock up at load time without a keyed diskette (such as its own) in one of the drives. The key disk may then be removed to free the drive.

Padlock is economical. You buy software that fingerprints 100 disks and their programs for $2.50 each, not costly fingerprinted disks you may never use at maybe $6.00 each.

Tell you what. PadLock keeps track of how many disks it has protected. Try it out. Format up to 5 test disks. If PadLock doesn't suit you, return it within 30 days for full refund.

Incidentally, don't look elsewhere. PadLock is available only from PC Brand.

Code	Our Price:
M0100 100 formats	**$250**

BRIEF

OUTSTANDING!

Is Anything But. A Whopper of an Editor

Dvorak in *Infoworld* said it for us: "the word is getting around...that Brief, The Programmer's Editor is simply the best text editor you can buy." With a name that belies its thoroughness, Brief™ has every feature you've ever contemplated for your editor-in-chief. Here goes:

Text, from keyboard or files, is housed in multiple buffers, and scrolled through one or more windows you open, close, resize. A text buffer may be called to different windows to view two areas at once. A change in one changes both. Text blocks may be marked for printing, writing to files, movement to scrap buffers for cut and paste into other buffers, or deletion, with as many "undo" levels as you want.

To find and fix, Brief has text search abilities rivaling "grep," with wild cards for matching, indifference to intervening characters, acceptance of character ranges, even multiple choice of patterns and their replacements.

If you use Lattice, C86™, or Wizard, and have 320k, you can compile your C program without ever leaving Brief. In fact, Brief will then find the lines with errors, and march you through the text for repairs.

On to macros! Parts of Brief were written with its own Lisp-like macro language. It has structure, 32-character variable names, conditional execution, loops, and you can actually read it! Nothing like the hieroglyphs we've seen elsewhere. A compiler comes with your copy, and prewritten macros in source to give you the flavor. Among the notables: one automatically indents C text; another turns Brief into a word processer with margin setting and wraparound.

There's a bulletin board with public domain macros contributed by a mushrooming cult of macromaniacs, and we've even hearing of megamoney prices for magnificance.

Then there's ... but, really, we're out of space. Who named this product, anyway?

Product Code: U0590(— Our Price: **CALL**
List Price: **$195**
Needs 192k

FIRSTIME

Assistant Programmer for Hire. Real Cheap.

Not interested in interpreters? Then explore the other fast lane to coding: language-specific editors. Among them, FirsTime™ is the strongest C editor we have seen, with many more commands and built-in utilities.

FirsTime asks what program structure you want and sets it up for you. It could be "main", or "function", or "if-else" or "while" or any component of C. Ask for it and FirsTime fills in its skeleton, tending to semi-colons and curly braces.

It puts placeholders into these structures to prompt you to enter workable code—the initializers, conditionals and counters which govern an "if", for example. It checks that variables you enter have been declared, and highlights errors until fixed. Material can be pulled into ten buffers for movement elsewhere. Your last delete is held in one of them for "undoing". There's search and replace, writing of any marked block to a satellite file or the printer.

FirsTime can even transform one type of structure into another, such as "for" into "while" or "while" into "do".

Movement through the screen highlights the successive logical blocks of your program so that its structure is always apparent. And FirsTime can hide different levels of code detail to show you only the outer framework of your program.

FirsTime saves keystrokes to add speed, indeed, but more important, *it lets you* think at a higher level and makes a big contribution toward error-free code.

Product Code: E0460 — Our Price: **$229**
List Price: **$295**

LMK

A Unix-like "Make" Makes Light of System Building in Any Language

If you have ever built a complex system, you know the time loss and tedium of recompiling, rebuilding libraries and relinking modules because a snippet or two of code has changed. Batch files are no answer. You need batches of them to avoid redoing everything indiscriminately.

Instead, imagine making a change deep in a system, and simply telling Lattice's LMK™ to take over. No further thinking or keystrokes. LMK will rebuild your final product, however involved and complex, by doing just what is needed and no more.

How? You write a command file which expresses, bottom to top, all the elements comprising your system and all its dependencies: what gets compiled to make what object file using what options; what is built into libraries; what is linked into the final EXE file. Through the life of your system LMK keeps track of the last time every action was performed. Run LMK and, tracking each branch, it looks only for elements which changed later than a dependent element further along the branch, using date and time information in the file directory. Any source file newer than its object file, for example. Only those elements and their dependents are re-made. All other instructions are bypassed.

LMK does not care what programming language you use; it's not just for C. For that matter, LMK can apply to more than programming. It can be used for *any* set of tasks which can be accomplished through commands issued to the operating system.

Wherever your imagination leads you, LMK will find the shortest path to get the job done. Minimum time, minimum effort software.

Product Code: L2100 — Our Price: **$159**
List Price: **$195**

CODE SIFTER

Find the Fast Track for Your Program

You know it runs, but how fast? If it can't win, better not enter your program in today's horse race. Don't let it out of the stable until you've put it through its paces with our value-priced stopwatch.

Code Sifter finds the sluggish spots in your program. On its own, it will divide a COM or EXE file of any size into thirty-two equal partitions. Or specify the partition boundaries with addresses, or with symbols, if your linker has produced a symbol map. Then tell Sifter to run your program. It samples your object file at precisely timed intervals and counts how many times it finds the instruction pointer in each partition. Job done, it reports the number and percentage of hits in each.

You are in for some surprises when you discover how unbalanced the activity is likely to be, and that's why Code Sifter is so valuable. It profiles just where you can best spend time optimizing your code, or even converting to assembler subroutines.

Code Sifter has a number of monitoring options. You can tell it to include any combination of your program, DOS, and BIOS in its analysis. You can specify the sampling rate. You can tell Code Sifter the number of times to run a program, and between each run discard the less active ranges, and re-partition the hot spots, so that you zoom in ever-tighter on small areas of code. Right down to the last byte! Try it on the sample program that comes with your disk.

Code Sifter. It will give you the racer's edge.

Product Code: N3100 — Our Price: **$99**
List Price: **$119**

C-SPRITE

Lattice's Own Symbolic Debugger for Lattice® C

This versatile companion to your compiler gives you the best of both worlds for an out of this world price. Hand it a COM or EXE file produced by the Lattice C™ compiler (using the -d option) and C-Sprite will speak your language: your function names, your variable names, your data types, and the line numbers from your source code. At the same time you can get a close-up view of machine addresses and machine-coded instructions, if you want to scrutinize just what machinations the compiler (or an assembler) contrived.

You already know how to converse with C-Sprite™ if you are familiar with Microsoft's Debug. Lattice began with that well-known command language, and added to it considerably: You can work with data in hex, as you might expect, but you can also differentiate between C's data types to cause the debugger to treat addresses as strings, long integers, even pointers, etc., both in display and entry.

C-Sprite can set breakpoints using symbols or addresses. You can submit clusters of commands to be executed at the breakpoints, or set commands that execute until a condition is met.

C-Sprite even has macros—use your source code variable names in a macro to dump the contents of entire C structures, for example. And you can debug through one of the COM ports with a second terminal so as not to disturb your program's display screen. What's more, if you link with Plink86™, C-Sprite can even tackle overlays.

Program doctors will find plenty of implements to rummage through in this kitbag.

Product Code: L2300 — Our Price: **$145**
List Price: **$175**

CURSES

A Screen Management Interface to Swear By

Curses manages the screen of the IBM PC in the same fashion as the curses utility of Unix and similar operating systems. Use it to adapt programs which call Unix's curses functions for screen management, and need the equivalent library when moved to the PC for re-compilation. Or use it when creating software on the PC to assure that it is Unix compatible.

Curses is a library of eighty-four functions and macros which can keep any number of screen images in memory. A screen may be full or partial size, and any can be summoned to the physical screen at your programs's command.

Within a screen, Curses employs a vast function set to get characters, wrap lines, scroll, blank lines, highlight— virtually any conceivable tool needed to update the screen. The product supports color, and all four memory models. Its input functions give you control over whether to echo each character to a memory screen. In keeping with the terminal orientation of Unix curses, the physical screen is repainted (at high speed) only when your program calls a refresh function.

Writing screen management code leads to unspeakable snarls and expressions. Swear off! Let Curses clean up your language.

Code:	List Price:	Our Price:
L0850	$125	**$100**
L0860 with Source	$250	**$200**

PC BRAND, Craftsman, PadLock and Toolbox TMs of PC BRAND / Unix TM Bell Laboratories / Lattice registered TM; and Lattice C, LMK, C-Sprite, CVUE and dBC TMs of Lattice Inc / dBASE TM Ashton-Tate / c-tree TM FiarCom / Pre-C, Plink86 TMs Phoenix Computer Products /MS TM of Microsoft / IBM registered TM Int'l. Business Machines / FirsTime TM Spruce Technology Corp. Inc. / WordStar registered TM Micropro / C86 TM Computer Innovations Inc / Access Manager, TM Digital Research / Brief TM UnderWare Inc.

EXPORT EXPERTISE

PC BRAND ships anywhere. We'll prepare the export documents and ship to you or your agent by air parcel post, air freight or courier. Phone or Telex your order for speed. To pay by credit card, please provide number, date of expiry, name and billing address of card. Or wire funds to PC BRAND, c/o Chemical Bank, 126 East 86th St., New York, N.Y. 10028, Account No: 034-016058. We will ship immediately and acknowledge by Telex if a number provided.

CRAFTSMAN™ QUALITY SOFTWARE FOR 'C' & OTHER PROGRAMMERS

C-TREE

NEW FEATURES!

B-Tree File Manager, Source Code, No Royalties!

A b-tree can be infested with bugs, so before buying one, ask its age. In a stand of saplings, this one is a real cequoia. C-tree™ has been around since 1979. (It became Digital Research's Access Manager™). That means seasoned, sturdy code which hasn't cracked under prolonged and widespread use.

C-tree comes in C source code, revealing all you've ever wanted to know about how b-trees are written. Provided you bind it into your binary application, you can re-distribute c-tree without royalties.

And if all this is disappointing, now the good part. C-tree's design splits nodes to allow any number of users to access an index file simultaneously even when updates are in progress so that multi-user configurations and adaptation to networks are possible.

The latest version has new features: support of variable record length data files; multiple key indexes in a single physical file; MS-DOS and Unix record locking examples.

Thanks to source code which does not deviate from K&R, C-tree can travel. Binary has always meant finding a substitute file manager for yet another compiler, operating system, or computer; then changing all the function calls program-wide; then starting the whole testing process anew. That's over. Tests in many environments prove that C-tree gives your application a ticket to anywhere.

C-tree permits any number of keys for a data file, supports duplicate keys, alphanumeric or numeric, etc., etc.: it's a big product with everything you'd expect. Beyond that it is intelligently designed as both a high level set of ISAM routines to minimize your coding by handling all details of adding a record on its own, for example; and as low level operations which you can access directly. Either way C-tree maintains optimal index structures which will find a record amongst a million ten byte keys in no more than five disk seeks.

Product Code: F0660‹	Our Price:
List Price: $395	$329

TEXT TOOLBOX

Tackles Text Tangles

Unix™ boasts a number of muscular utilities that are migrating to the PC world. Lattice has assembled a cluster of the most useful text management tools into a single package.

"Grep" looks for text patterns in any number of files. Want all occurrences of a global variable throughout a program system? Want to search all programs in a directory, down paths to other directories, or all files on a disk? Need to find all the function calls in an entire program system? Grep can do it with a powerful expression syntax that goes far beyond your text editor's search command.

"Ed" is similar to the well-known Unix editor. It offers search and replace with "grep's" syntax, block move, read and write, optional line numbering, append, insert, delete, and this unusual facility: you can instruct "ed" to apply a file of commands to any number of target files, even complicated changes and text additions, such as those created by "diff".

"Diff" compares text files line for line. Its output is a precise list of instructions telling what to do to make two files the same, a list which can be handed to "ed" to do.

Code:	List Price:	Our Price:
L2200	$120	$100
L2205 with Source	$240	$200

CVUE

Make Your Own Editor

CVUE is a low-priced screen-oriented text editor which does most of the things that a good editor should do, and boasts full DOS 2.0 directory path name support in reading and writing files.

It was written by the Lattice programmers who felt forgotten by the folks who write WP software. They needed easy entry of non-display characters such as control codes and escape sequences, not footnotes; indenting and undenting, not italics; pattern searching, not spell checking. So CVUE™ was born.

CVUE only supports in-memory text files, but with memory at today's prices, creating and maintaining files of over 500 K is practical. As compensation, CVUE is very compact and fast. It actually runs in computers with only 64 K of memory and uses no tediously slow overlays.

The power of CVUE is its ease of customization. And when you take advantage of the Source Code option, the resultant editor can be made truly your own.

Code:	List Price:	Our Price:
L2240	$ 75	$ 69
L2245 with Source	$200	$220

USED COMPILERS WELCOME

Trade In for the Latest Model Lattice C

Has your compiler run out of gas? Has your model been discontinued? Is it falling behind for lack of new parts?

Even if yours is in good shape, you have surely noticed there are more options and accessories produced to run with the Lattice C compiler than any other. Don't do without these additives any longer. It's time for new license plates. Trade in your original disks and manual of any of the compilers below and we'll send you Lattice's most up-to-date model.

From then on you will be adopted by Lattice for full, direct support by their technical specialists.

	Price:
Microsoft MS-DOS/PC-DOS C	$150
Computer Innovations C86, Mark Williams C, Digital Research C, Whitesmith's C	$200

PRE-C

Thorough "Lint"-like Analysis Now on the PC

Unix users long for a "lint" to give programs a thorough cleaning before they disappear into a compiler.

Pre-C™ looms larger than "lint." It finds problems your compiler won't. Problems that a debugger will have trouble figuring out. Even problems which will cause trouble with other compilers.

Pre-C finds all the syntactical tripwires that will blow out a compiler, sure, but it goes after subtler problems: code which will never be accessed, casts with suspect conversions, variables declared as external but never used, functions never called, obsolete usage (even C has changed), machine-dependent expressions with will inhibit portability.

Compilers work with one module at a time. They know nothing of other modules which only meet up at link time. Pre-C can look at *all segments of your program at once* and report to you any inconsistencies of inter- module references: conflicting data type declarations, parameter lists in function calls which disagree with the functions themselves in number or data type, declarations of external functions which differ from their definition.

Pre-C uses the Unix System III compiler standard to safeguard maxium portability anywhere in the C world. There are then plentiful command line options to advise Pre-C what to flag and what to forgive, useful during early coding when some functions are empty or incomplete. The output of each analysis can be filed for use with subsequent Pre-C runs, so work is not performed redundantly.

Pre-C lets you develop standing profiles of binary libraries. In any C program you subsequently write, Pre-C can use these profiles to make sure your calls to those libraries' functions are perfect.

This is a big product which will work miracles in speeding large system development. 128K minimum; 192K recommended.

Product Code: P0590	Our Price:
List Price: $395	$329

30-DAY MONEY-BACK GUARANTEE

We will return the purchase price[a] of any product in this space if you decide it fails to meet your needs except those coded "‹" by requirement of their suppliers. We want you to be completely satisfied with your purchase. Go through the manuals, *try out the products*, and make a thoroughly informed decision whether they are right for you.

The only requirement for refund or credit is that we must *receive* your returned product within 30 days of its shipment, and it must in our judgment be in 100% resaleable condition.

[a] Not shipping and handling.

SUPPORT DIRECT FROM LATTICE

Products with codes beginning "L" are all fresh from Lattice Inc., the premier software developer serving the C professional marketplace. PC BRAND has made special arrangements for Lattice to provide telephone support for these products directly.

Take advantage of this unusual opportunity. Buy from PC BRAND, and you get to talk to the experts themselves!

dBC

Switch from dBASE Language to C for Power, Speed

There are a lot of dBASE™ file users out there. Most of them just keep data bases and use dBASE's limited reporting facilities. They're not programmers, so they don't use the dBASE programming language. But they'd like more for their efforts, and that's a business opportunity.

dBC™ links C to dBASE. It creates and maintains files and their indexes which exactly replicate dBASE file design. So dBASE can read and update them. And the reverse. dBC can use any files created by dBASE. Now C and dBASE can operate on the same data bases interchangeably.

That opens up the widespread culture of dBASE installations to exploitation by C programmers. Now you can replace the resident dBASE language with the speed of C. And you no longer have to write every line of code, as in the dBASE language, because now you've unlocked C's vast storehouse of off-the-shelf libraries and utilities.

dBC's functions parallel all dBASE's file handling commands, many of them decomposed to give you closer-in control. The manual discusses each in detail, and demonstration source files on your disk show how every function is used. Use dBC for custom work for clients, or design generalized programs for manipulation and reporting of dBASE data bases.

Or use dBC on its own. It's a complete ISAM file manager for use with the Lattice C compiler whether or not dBASE will ever be used in tandem, has versions for all four memory models, and can have sixteen index and data files open at once.

Code & Version:	List Price:	Our Price:
L00II: Dbase II Compatible	$ 250	$225
LCCII: with Source Code	$ 500	$450
L0III: Dbase III Compatible	$ 250	$225
LCIII: with Source Code	$ 500	$450

TERMS AND CONDITIONS OF SALE

Licenses: Each price is for a license to use a product on a single computer and does not constitute its ownership. We will inquire for you about availability of multiple machine licenses at a single site. Except for those coded "#", products may be used to create programs for distribution without royalty payments or additional licenses, provided said programs do not substantially replicate the products themselves.

Compatibility: PC BRAND's standard products are designed to operate with the IBM* PC, XT or AT under PC-DOS and require no more than 128K of RAM unless indicated. Non-IBM machines using MS-DOS: If in doubt as to total compatibility, please determine from manufacturer.

Returns: Defective parts will be replaced. Products coded "‹" are rendered unreturnable if sealed envelopes containing diskettes are opened. Otherwise, to return a product for refund or credit, call (212) 410-4000 for required prior authorization.

Payment: We honor MasterCard, Visa, American Express (no surcharge), wired funds, checks in advance (in NY State, add sales tax), and will ship COD (U.S. only) for cash, money order, or certified check (no fee). Purchase orders accepted from larger corporations and institutions at our discretion; terms net 30, 2% a month late penalty.

Shipping & Handling: U.S.: UPS Surface: 1st product $6, each add'l $3. UPS 2nd Day Air: 1st product $10, each add'l $4.50. UPS Next Day Air or Federal Express 1-2 Day Air: 1st product $18, each add'l $6. Federal Exp. Next Morning: 1st product $28, each add'l $7. International: Charges depend on destination and shipping method. $10 per shipping container for export or customs forms.

To Order Products or Catalog, Call Us at...

800-PC BRAND

That' (800) 722-7263. In NY State or Outside U.S. call (212) 410-4000.
PC Brand, P.O. Box 474, New York, N.Y. 10028
TELEX: 667962 (SOFT COMM NYK)

CIRCLE NO. 171 ON READER SERVICE CARD

Prices, terms, and specifications subject to change without notice.

SAVE $5
Just tell us this code:
J10

PERSONAL COMPUTER PHOTOCOPIER

Have you ever wished that there was a machine for your PC that could scan your photographs, artwork or documents just like a photocopier? And was as easy to use and understand? As well as copy and store in color or black-and-white onto your hard drive for editing?

Well, there is such a machine now. It's called the **SpectraFAX** digital photocopier. We got tired of trying to use digitizing tablets, cameras, mice and everything else that has been invented. The **SpectraFAX** machines will scan any and all of your art, photos, forms or text for storage in your PC. Images that you can cut-and-paste into your word processing or database programs. Our **SpectraFAX 200** will scan and store your color images for editing and printing on your color printer at resolutions up to 200 dots per inch. The **SpectraFAX DS-200** will do the same for you in black-and-white. Then the **SpectraFAX Graphics Editor** software that we bundle with the scanner will allow you to cut-and-paste your images, rotate them, enlarge and reduce whatever pictures you choose.

That's not all. Our digital photocopiers are based on open-architecture design, which means that as new uses for this technology develop, add-on cards can enhance our **SpectraFAX** digital photocopier. But we didn't want to wait for third-party vendors, so we invented two optional boards of our own: the **SpectraFAXimile Card** and the **SpectraFAX TEXreader.** The **SpectraFAXimile card** takes advantage of the 200 dots per inch resolution of the scanning device to turn it into a facsimile machine. The **TEXreader** board is a revolution in itself — full-scale OCR (optical character recognition) to read typewritten pages with the **SpectraFAX** machine into your word processor without having to re-type what somebody else has already typed.

What's most amazing about the **SpectraFAX** product line, though, are the prices. Our basic black-and-white digital photocopiers cost about the same as a regular office copier (which you may never use again!) and our color scanners cost very little more. Because we know high tech has to be affordable.

- IBM is a registered trademark of International Business Machines, Corp.
- SpectraFAX, SpectraFAX 200, DS-200, SpectraFAX Graphics Editor, SpectraFAXimile & TEX reader are trademarks of SpectraFAX Corp.

SpectraFAX Corp.

2000 Palm Street S. • Naples, Florida 33962 • (813) 775-2737

CIRCLE NO. 194 ON READER SERVICE CARD

A Good Find

This program gives the user a direct access to all hard-disk directories and subdirectories in finding a lost file.

MARK S. ACKERMAN

Late Friday afternoon, the boss asks for five copies of the Auburn letter (the Auburn account is, of course, the most obscure account the company has). With a sinking feeling, you realize you have no idea where the letter is. After searching among all the directories on all the hard disks in the office, you decide there must be a better way.

Files can become misplaced easily on high-capacity hard disks, leading to situations such as the above. The problem is usually a matter of not remembering where a file was stored or under what classification. DOS has no facility for finding such a file and the only answer is to search patiently through every possible subdirectory. The Where program (listing 1) solves this problem.

The following simple command

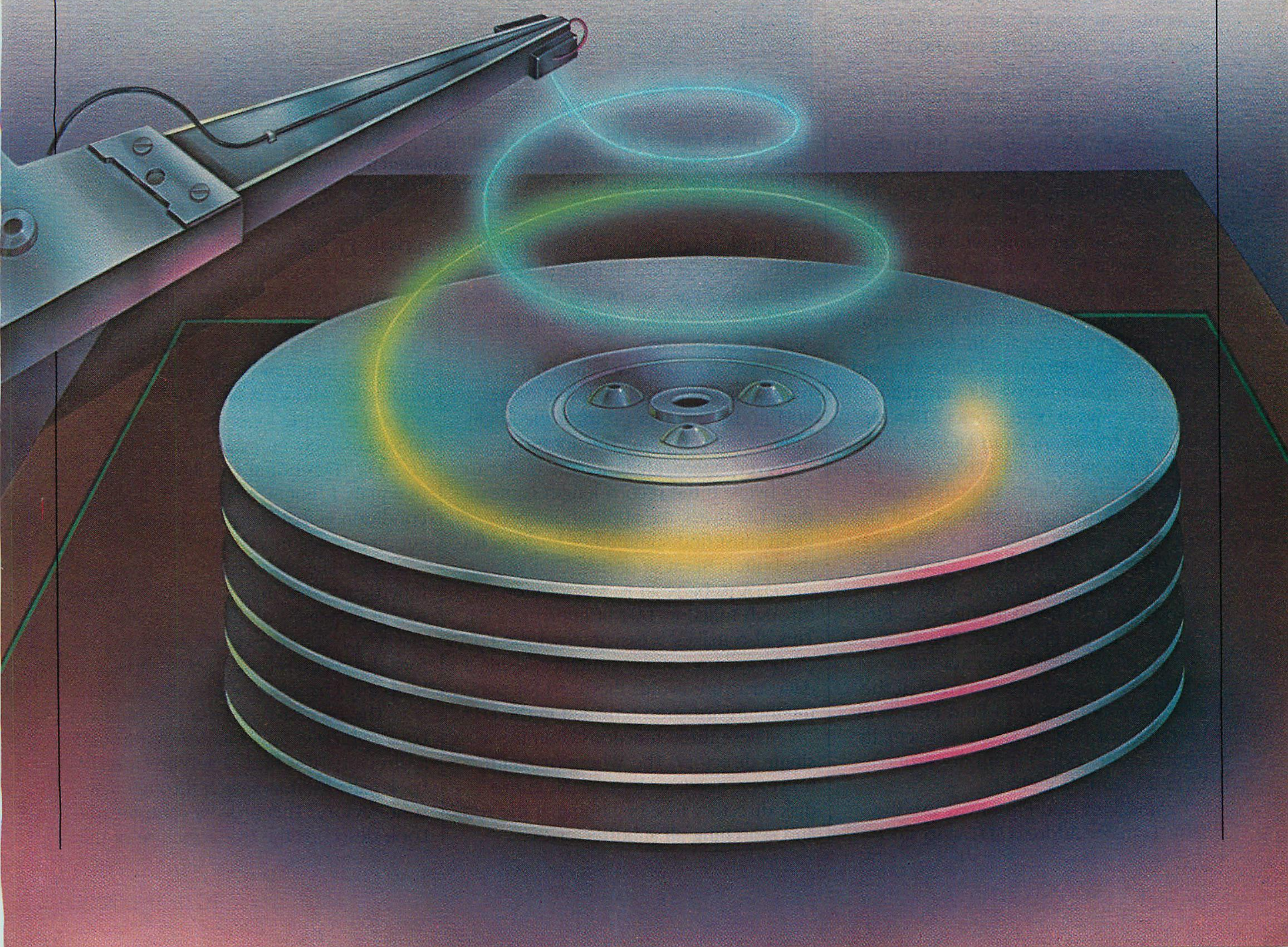
```
WHERE auburn.1tr
```

starts the Where program, which quickly (and patiently) searches a hard disk for the file. DOS wild cards are accepted; therefore, the command

```
WHERE auburn.*
```

would instruct the Where program to find all occurrences of files with the name Auburn. Where can be run from any subdirectory and will find files in any subdirectory. It returns a list of file locations, sizes, and dates that would be similar to the sample screen print for this example, shown in figure 1.

ILLUSTRATION • ANDY LEVINE



The core of the program is two DOS interrupt 21H function calls. The Find First function (4EH) finds the first file that matches the user input. The Find Next function (4FH) finds the next matching file. Together these two functions can form a loop to search for all the matching files in a directory.

Find First requires that the DS:DX register pair point to an ASCII string specifying the drive, path, and file name. The file name can contain the usual DOS global search characters * and ?. The ASCII string must be terminated with a binary zero, as is usual with both DOS and C (the language in which Where is written)

In addition, Find First requires that the CX register contain file attributes. DOS has five types of file attributes: read-only, hidden, system, directories, and archive. Each takes a bit on the file attribute byte (see table 1). The use of the attribute bits is generally straightforward: A program cannot write to any file that has the read-only bit set in its directory entry. A file with the hidden or the system bits set does not appear in a DIR listing. DOS sets the archive bit whenever a file is written to and closed; the bit is cleared whenever a BACKUP is done on the file. A normal data file can have the archive bit either set or clear, depending on when the file was last backed up.

However, doing a search using file attributes is not straightforward. Searches performed with the archive bit or the read-only bit set, or with no bits set, will all return the same set of files. That is, no matter how it is done, a search with the Find functions will always return normal files, with and without the archive bit set, and any read-only files.

Searches with the hidden, system, or directory bits set are *inclusive*—a directory will yield normal files with and without the archive bit set, any read-only files, and subdirectories. This behavior is consistent from DOS 2.1 to DOS 3.0; however, the error codes returned are not. When an error condition occurs, the carry bit is set and an error code is returned in AX. The *DOS 2.1 Technical Reference* states that the error codes returned are 2 (file not found) and 18 (no more files) for Find First and 18 for Find Next. The *DOS 3.0 Technical Reference* does not state which error codes are returned. If the search string handed to Find First is incorrectly formed, DOS 3.0 will actually return error code 3 (path not found). DOS 2.1 returns code 2.

Find First and Find Next use the Disk Transfer Address (DTA) to store information on the found files. The DTA is filled in as shown in table 2. Bytes 0 through 20 are set by Find First. They are then used by all subsequent Find Next function calls.

DOS normally uses the DTA as a buffer for file I/O, placing it at offset 80H in the segment prefix at the beginning of the program. This location, however, effectively limits the buffer to 128 bytes. If the record length is more than 128 bytes, the applications program must find some other location for the buffer. Because of this, DOS allows the DTA to be reset to any location through function 1AH of interrupt 21H. This also makes it easy to reset the DTA to the location of a C structure. Using a data structure avoids the need for parsing the information returned in the DTA.

These three functions examine a single directory. The Where program, however, needs to look at the entire disk directory tree beginning at the root and searching all subdirectories.

LOOK IT UP

Where uses recursion, an operation ideally suited to tree searches, to examine the directory system of the PC. A recursive function is defined by two criteria: it can call itself and it must have some ending condition. A factorial function is a good example: one way to solve n! is to successively find the solution to (n - 1)!. The ending condition is that 0! equals 1. For Where, the ending condition is that there are no more subdirectories in a directory.

A critical characteristic of recursive functions is that they return to the point where they left off. All local variables are restored to the values they had before the recursive call; global variables keep any changes that have been made.

The recursive portion of Where is Look, which is called to search through a directory. If it finds a subdirectory, Look calls itself to search through that subdirectory. This process continues as Look descends the tree structure of the

FIGURE 1: *Sample Screen Print*

```
C>where auburn.*

     3328  10-16-84  03:25 PM  \contract\auburn.2
     1139  10-20-84  02:24 PM  \contract\auburn.1
    12928  02-11-85  10:54 AM  \ack\admin\current\auburn.wks
     1408  02-28-85  12:53 PM  \ack\admin\current\auburn.doc
     1280  03-04-85  06:16 PM  \ack\archive\auburn.ltr
```

The above is a sample list of file locations, sizes, and dates that might be returned by the Where program in answer to the command WHERE auburn.* Where can be run from any subdirectory; it uses recursion to find files in any hard-disk directory.

TABLE 1: *File Attribute Byte*

BIT	ATTRIBUTE
00H	Normal data file—archive bit not set
01H	Read-only file
02H	Hidden file
04H	System file
10H	Directory
20H	Normal data file—archive bit set

Find First requires that the CX register contain file attributes. DOS has five types of file attributes, each of which takes a bit on the file attribute byte. Although their use is generally straightforward, the situation changes when they are used in searches.

TABLE 2: *Disk Transfer Address*

BYTES	STORED INFORMATION
0 - 20	Used by the next Find Next call
21	File attribute
22 - 23	File time of creation
24 - 25	File date of creation
26 - 29	File size (low word first)
30 - 42	File name and extension followed by a zero byte

The Disk Transfer Address (DTA) is filled in as shown above. Bytes 0 through 20 are set by Find First and are used by all subsequent Find Next function calls. Find First and Find Next use the DTA to store information on the found files.

directory system. When Look finds no other subdirectories, it looks through the normal files for a match and then returns. This is its ascent back up the tree structure, returning to the parent directory of the one it just searched.

Look examines every possible subdirectory and file for matches to the search string given in the command line. Every time a match occurs, Where prints the file information. If only a file name is given in the command line, Where begins the search from the root directory and moves through the entire system. However, if a directory precedes the file name, as in

WHERE \letters \auburn.???

then only subdirectories of the given directory are searched. In this case, only letters and its subdirectories would be examined for files matching auburn.???

The Where program, written in Mark Williams MWC86, uses three common C functions: **rindex** finds the last occurrence of a character in a string and returns a pointer to (the address of) that character; **strcpy** copies the second string named in the parameter list to the first string; and, finally, **strcat** concatenates the second string onto the end of the first string.

This C also has a bridge, called **intcall**, to the DOS interrupts that allows programs to call DOS functions. **Intcall** requires that the values requested in any registers be sent in a structure of type **reg**. It then places the values in the registers and calls the interrupt. After the interrupt, **intcall** places the register values into the **reg** structure specified. In the listing, the structure is called REGISTERS. The **ptoreg** function merely places a pointer value (an address) in a register pair.

Two points are crucial to a successful implementation of Where. First, within Look's main loop is a check to discover whether the current file (in the DTA) is a directory and whether the first character of that file is a period. This check may seem superfluous, but remember that Find First and Find Next both return not only directory names but also normal file names. (This is how the change in error codes from DOS 2.1 to 3.0 became apparent.) In addition, the program wants to search only valid subdirectories. The two subdirectory names, . (the current directory) and .. (the parent directory), are not valid subdirectories for searching.

Second, because the facilities of a high-level language restore the values of the local variables after a recursive call, it might seem that the DTA need not be reset immediately after the call to Look inside of Look. This is partly true. The address of the current DTA data structure is correct: the current DTA goes with the current level of recursion. However, while Where knows which DTA to use, DOS does not. DOS has not been part of the recursion, so the current DTA must be reset inside DOS with another Set DTA call.

When compiling Where, be sure to allow between 2KB and 5KB of stack space for recursion. The program itself is 8KB. Also note that increasing the number of buffers in the CONFIG.SYS file can greatly impact Where's execution time. For example, a PC/XT with 6.3MB memory took longer than one minute to do a search using two buffers; with 16 buffers, the time was reduced to about 12 seconds.

Where could be modified to include better output and expanded error messages. The listing could include only the first instance of the directory name, indenting for others. Error messages that indicate, for example, when no parameters have been entered in the command line or that no matching files were found would be helpful. The most useful addition might be a command line flag to signal that only the first match is wanted. Then the user would not have to abort after the first file name is printed.

Mark S. Ackerman is vice-president of engineering at Ten Point Systems. He has a master's degree in computer science.

LISTING 1: WHERE.C

```
/*****************************************************************
*                                                                *
*        WHERE                                                   *
*                                                                *
* Where is a program to locate files on the PC hard disk.        *
* It requires DOS 2.x or 3.x.                                    *
*                                                                *
* The command line syntax is:                                    *
* where [starting directory]filename.ext                         *
*                                                                *
* Written by Mark S. Ackerman                                    *
*                                                                *
* Copyright 1984, 1985 by Mark S. Ackerman.  Permission is       *
* granted for unlimited copies if not sold or otherwise          *
* exchanged for gain.                                            *
*                                                                *
*****************************************************************/


/*****************************************************************
* The C header files                                             *
* These identify library routines like printf() and intcall()    *
*****************************************************************/

#include <stdio.h>      /* standard i/o                          */
#include <dos.h>        /* functions for DOS interrupt calls     */

/*****************************************************************
* Definition of DOS Disk Transfer Area (DTA)                     *
*****************************************************************/


/*****************************************************************
* Structure for MS-DOS date and time fields                      *
* See pages 4-6 and 4-7 of the DOS 2.1 technical                 *
* reference manual for more information                          *
* This structure is used in the next structure definition        *
*****************************************************************/
struct msdos_date
        {
        unsigned ms_sec    : 5; /* time in 2 sec. int (5 bits)*/
        unsigned ms_min    : 6; /* minutes (6 bits)           */
        unsigned ms_hour   : 5; /* hours (5 bits)             */
        unsigned ms_day    : 5; /* day of month (5 bits)      */
        unsigned ms_month  : 4; /* month (4 bits)             */
        unsigned ms_year   : 7; /* year since 1980 (7 bits)   */
        };

/*****************************************************************
* Structure filled in by MS-DOS for interrupt 21 calls           *
* See page 5-46 of the DOS 2.1 technical reference               *
* manual for more information                                    *
*****************************************************************/
struct DTA
        {
        char    DTA_dosinfo[21];    /* used by DOS            */
        char    DTA_attr;           /* file attribute byte    */
        struct msdos_date DTA_date; /* date struct. as above  */
        long    DTA_size;           /* file size              */
        char    DTA_filename[13];   /* file name (w/o path)   */
        };

/*****************************************************************
*                  Definitions of constants                      *
*****************************************************************/
#define not !      /* for ease of reading (C uses a !
                      character for logical not's)            */
#define and &      /* for ease of reading (C uses a &
                      character for bit-wise logical and's)   */
#define carry_set 0x0001 /* mask for flag register
                            for carry bit      */
#define no_type   0x00   /* no bits set on file attribute byte*/
#define directory_type  0x10 /* directory file bit on file
```

```
                                  info word       */
#define no_more_files    18      /* DOS return code for
                                    no more files         */
#define end_of_string    '\0'    /* C uses a binary zero to
                                      signal end of string    */
#define backslash '\\'   /* the backslash character       */

char *month[] = {
              "Jan","Feb","Mar","Apr","May","Jun",
              "Jul","Aug","Sep","Oct","Nov","Dec"
              };
char *time_of_day[2] = {"AM","PM"};

/*************************************************************
*              Define the type "filename"                    *
*                  to be a string of 51 characters           *
*************************************************************/
typedef char filename[51];

/*************************************************************
*                                                            *
* The following filename strings are used in the program:    *
*                                                            *
*        check_string            filename to be searched for *
*                                filename in the command line) *
*        directory_string        directory name to be searched *
*        newdirectory_string     directory name to be searched *
*                                  on next recursive call    *
*        current_string          temporary string for searching *
*                                  in a specific directory   *
*************************************************************/

/*************************************************************
* Definition of any forward-referenced functions             *
*************************************************************/
char *DATE();

/*************************************************************
*              Global variables                              *

*************************************************************/
filename check_string;  /* this string "remembers" user input */
struct reg registers;   /* structure to allow access to indiv.*/
                        /*   registers for interrupts        */
char datestring[40];    /* print output string for dates     */

/*************************************************************
*              MAIN() -- the beginning of the code           *
*************************************************************/
main(argc,argv)
int argc;
char *argv[];
    {
    filename directory_string;  /* directory to be searched    */
    char *incoming_filename;    /* address of filename in
                                   command line argument
                                   (ie, the filename)          */
    char *last_location;        /* address of last backslash in
                                   command line argument       */
    char *incoming_string;      /* address of
                                   command line argument       */
    int  last_directory_char;   /* last character
                                   in directory string    */

/*******************************************************
*       check number of incoming arguments              *
*       if incorrect, write an error message            *
*******************************************************/
    if (argc not= 2)
        printf
    ("usage is:   WHERE [starting directory]filename.ext\n\n");
    else
    {

/*******************************************************
* incoming_string is set to the first argument in the   *
* command line                                          *
* The incoming_string is then searched for the last     *
* occurrence of a backslash to find the end of          *
* the directory name.                                   *
*******************************************************/
incoming_string = *(++argv);
last_location = rindex(incoming_string,backslash);

/*******************************************************
* If there was not a backslash (and therefore the       *
*     beginning directory is the root directory)        *
* begin                                                 *
*   copy command line argument into check_string        *
*   copy root directory into directory_string           *
* end                                                   *
* else                                                  *
* (if there was a backslash and therefore a beginning   *
*      directory specified in the command line)         *
* begin                                                 *
*   set the incoming_filename to the next character     *
*        past the backslash                             *
*   copy the incoming_filename into check_string        *
*   copy the command line argument into                 *
*        directory_string                               *
*   terminate directory_string just after the           *
*        last backslash (therefore leaving only the     *
*        the directory name in the string)              *
* end                                                   *
*******************************************************/
if (last_location == NULL)
    {
    strcpy(check_string,incoming_string);
    strcpy(directory_string,"\\");
    }
else
    {
    incoming_filename = last_location + 1;
    strcpy(check_string,incoming_filename);
    strcpy(directory_string,incoming_string);
    last_directory_char = incoming_filename - incoming_string;
    directory_string[last_directory_char] = end_of_string;
                                    }

/*******************************************************
*                start 'er up                           *
*******************************************************/

LOOK(directory_string);
}
return;
}

LOOK(directory_string)

/*******************************************************
*       LOOK is the recursive procedure in WHERE        *
*       It is called once for each subdirectory         *
*******************************************************/

char *directory_string;
    {
    struct DTA current_DTA;  /* used to return data from DOS  */
    filename newdirectory_string; /* the directory to be
                                    searched on the next
                                    call to LOOK()          */
    filename current_string;     /* temporary filename
                                    string for searching for
                                    directories             */

/*******************************************************
* Form current_string by copying directory_string and   *
*    and then concatenating "*.*" to look through all   *
*    files                                              *
*******************************************************/

    strcpy(current_string,directory_string);
    strcat(current_string,"*.*");

/*******************************************************
* Set the Disk Transfer Area in DOS to the current_DTA  *
*    structure                                          *
* Get the first subdirectory in this directory          *
```

```
*********************************************************/

    SET_DTA(&current_DTA);
    GET_FIRST(current_string,directory_type);

/*********************************************************
* while there are more subdirectories in this directory *
* begin                                                 *
*   double check for proper directories (see text)      *
*   if a directory                                      *
*   begin                                               *
*     set up the newdirectory_string for the            *
*     next call to LOOK (see text)                      *
*     call LOOK                                         *
*     reset Disk Transfer Address (see text)            *
*   end                                                 *
*   look for next directory                             *
* end                                                   *
*********************************************************/
    while (not(registers.r_flags and carry_set))
        {
        if (current_DTA.DTA_attr == directory_type &&
            current_DTA.DTA_filename[0] not= '.')
            {
            strcpy(newdirectory_string,directory_string);
            strcat(newdirectory_string,current_DTA.DTA_filename);
            strcat(newdirectory_string,"\\");
            LOOK(newdirectory_string);
            SET_DTA(&current_DTA);
            }
        GET_NEXT();
        }

/*********************************************************
* if there are no more subdirectories in this directory *
*   look for files                                      *
* else                                                  *
*   print an error message                              *
*********************************************************/
    if (registers.r_ax == no_more_files)
        GET_FILES(directory_string,&current_DTA);
    else
        printf("problem with looking thru %s\n",directory_string);
    return;
    }

GET_FILES(directory_string,current_DTA)

/*********************************************************
* GET_FILES                                             *
* is called once per directory to look for the          *
*   actual files matching the search string             *
*********************************************************/

char *directory_string;
struct DTA *current_DTA;
    {
    filename current_string;

/*********************************************************
* Form current_string by copying directory_string into  *
*   it and then concatenating the check_string onto     *
*   the end                                             *
*********************************************************/
    strcpy(current_string,directory_string);
    strcat(current_string,check_string);

/*********************************************************
* Get the first file that matches current_string        *
*********************************************************/
    GET_FIRST(current_string,no_type);

/*********************************************************
* while there are more files that match the search      *
*   string:                                             *
* begin                                                 *
*   print the file information                          *
*   get the next file                                   *
* end                                                   *
```

```
*************************************************************/
    while (not(registers.r_flags and carry_set))
        {
        printf(" %10D  %s  %s%s\n", (*current_DTA).DTA_size,
            DATE(&((*current_DTA).DTA_date)), directory_string,
            (*current_DTA).DTA_filename);
        GET_NEXT();
        }

/*************************************************************
* if error in looking for a file                            *
*    print error message and return                         *
*************************************************************/

    if (registers.r_ax not= no_more_files)
        printf("problem with looking for %s\n",current_string);
    return;
    }

GET_NEXT()
    {
/*************************************************************
* GET_NEXT does an interrupt 21h, function 4Fh              *
*************************************************************/

    registers.r_ax = 0x4f00;
    intcall(&registers,&registers,0x21);
    return;
    }

SET_DTA(current_DTA)
    struct DTA *current_DTA;
    {
/*************************************************************
* SET_DTA does an interrupt 21h, function 1Ah               *
*    The DS:DX pair is set to the address of the            *
*    current_DTA data structure                             *
*************************************************************/
    registers.r_ax = 0x1a00;
    ptoreg(dsreg,registers.r_dx,registers.r_ds,current_DTA);
    intcall(&registers,&registers,0x21);
    return;
    }

GET_FIRST(search_string,filetype)
    char *search_string;
    int filetype;
    {
/*************************************************************
* GET_FIRST does an interrupt 21h, function 4Eh             *
*    The CX register is set to either normal or             *
*    directory type (see text)                              *
*    The DS:DX pair is set to the address of the            *
*    search string                                          *
*************************************************************/
    registers.r_ax = 0x4e00;
    registers.r_cx = filetype;
    ptoreg(dsreg,registers.r_dx,registers.r_ds,search_string);
    intcall(&registers,&registers,0x21);
    return;
    }

char *DATE(dateptr)
struct msdos_date *dateptr;
    {
/*************************************************************
* DATE takes the date field from the current DTA            *
*    structure and returns a string containing the          *
*    information in formatted ASCII                         *
*************************************************************/
    sprintf(datestring, "%02d-%02d-%2d  %02d:%02d %s",
        dateptr->ms_month, dateptr->ms_day,
        dateptr->ms_year+80, (dateptr->ms_hour)%12,
        dateptr->ms_min, time_of_day[((dateptr->ms_hour)/12)]);
    return(datestring);
    }
```

Discover the most powerful

The IBM Personal Computer AT.

Hold on to your hat.

The IBM Personal Computer AT (for Advanced Technology) is based on the advanced 80286 16-bit microprocessor. This remarkable computer will run many of the programs written for the IBM PC, up to three times faster. You'll be able to recalculate large spreadsheets in seconds and retrieve files in a flash. And it's ideal for IBM TopView, the new kind of software program that lets you run and "window" several other programs at once.

The IBM Personal Computer AT has got the power (and price) to surprise you. In many ways.

Compatibility, expandability, networking too.

With the IBM Disk Operating System, the IBM Personal Computer AT can use many programs from the fastest-growing library in the personal computer software industry.

The IBM Personal Computer AT is also available with up to 3 million bytes of user memory to run multiuser, multitasking operating systems such as XENIX™. Volume upon volume of information is available at your fingertips. You can customize your system to store up to 20,000 pages of information at one time. And its keyboard helps you use all of this computing power more easily.

This member of the IBM PC Family is a powerful stand-alone computer that can also be both the primary file server and a station on your network. With the IBM PC Network (which is so easy to connect you can do it yourself), the IBM Personal Computer AT can share information with IBM PCs, PC/XTs and IBM *Portable* PCs.

IBM Personal Computer AT Specifications

User Memory
256KB-3MB*

Microprocessor
16/24-bit 80286*
Real and protected modes*

Auxiliary Memory
1.2MB and 360KB diskette drives*
20MB fixed disk drive*
41.2MB maximum auxiliary memory*

Keyboard
Enlarged enter and shift keys
84 keys
10-foot cord*
Caps lock, num lock and scroll lock indicators

Display Screen
IBM Monochrome and Color Displays

Operating Systems
DOS 3.0, XENIX,* PC/IX 1.1

Diagnostics
Power-on self-testing*
Parity checking*
CMOS configuration table with battery backup*

Languages
BASIC, Pascal, FORTRAN, APL, Macro Assembler, COBOL

Printers
Supports attachment of serial and parallel devices

Permanent Memory
(ROM) 64KB
Clock/calendar with battery*

Color/Graphics
Text Mode
Graphics Mode

Communications
RS-232-C interface

Networking
High-performance, high-capacity station on the IBM PC Network*

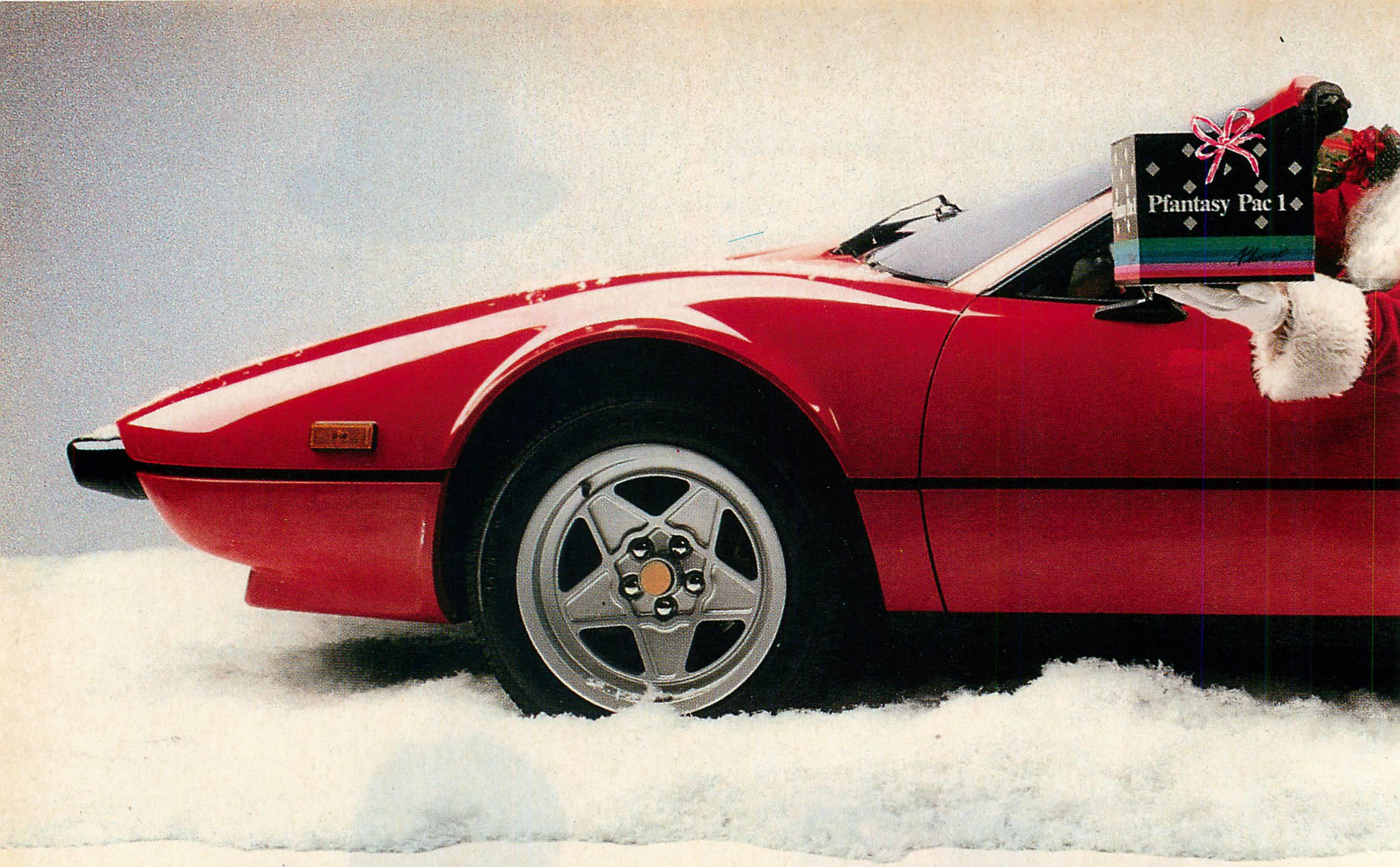
*Advanced Features for Personal Computers

Get a hands-on, hats-off demonstration.

The IBM Personal Computer AT has the power, compatibility and expandability many PC users need, at a very appealing price.

For more information contact your authorized IBM PC dealer, IBM Product Center or IBM marketing representative. For a store near you call 1-800-447-4700. In Alaska or Hawaii call 1-800-447-0890. IBM®

personal computer IBM has ever made.



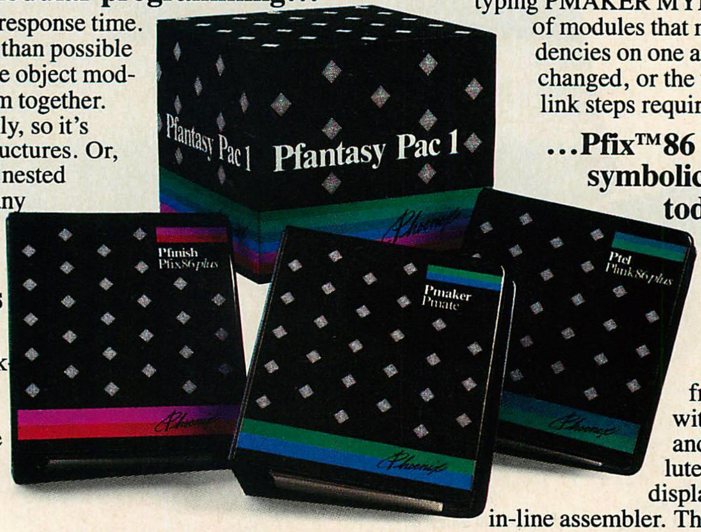
Buy Yourself A Pfantas

Just $~~12~~

Give your favorite programmer a holiday treat – Phoenix Pfantasy Pac1. Six of the best-engineered, highest performance programming packages on the market today. $1800 sold separately. $1295 as Pfantasy Pac1 with new software, and a unique on-line software update service. All at the one-time holiday price of $995. The perfect gift for the knowledgeable programmer. You'll get:

...Plink™86 Plus, our new linkage editor, adds a new dimension to modular programming...

Cache overlays to reduce response time. Link programs much larger than possible with standard linkers. Merge object modules so you can package them together. Reload overlays automatically, so it's easier to design program structures. Or, create up to 4,095 overlays, nested up to 32 levels, in one or many files, on one or many disks.



...Plus, Pmate,™ the text editor that works the way you do...

You can run Pmate in background while you perform another task, or call it up instantly while you're inside another program. With Pmate, you can reassign any command or key, so you can use keystrokes learned on other editors or make your keyboard Dvorak. And, you can assign complicated editing tasks to a keystroke, so you never have to re-key them.

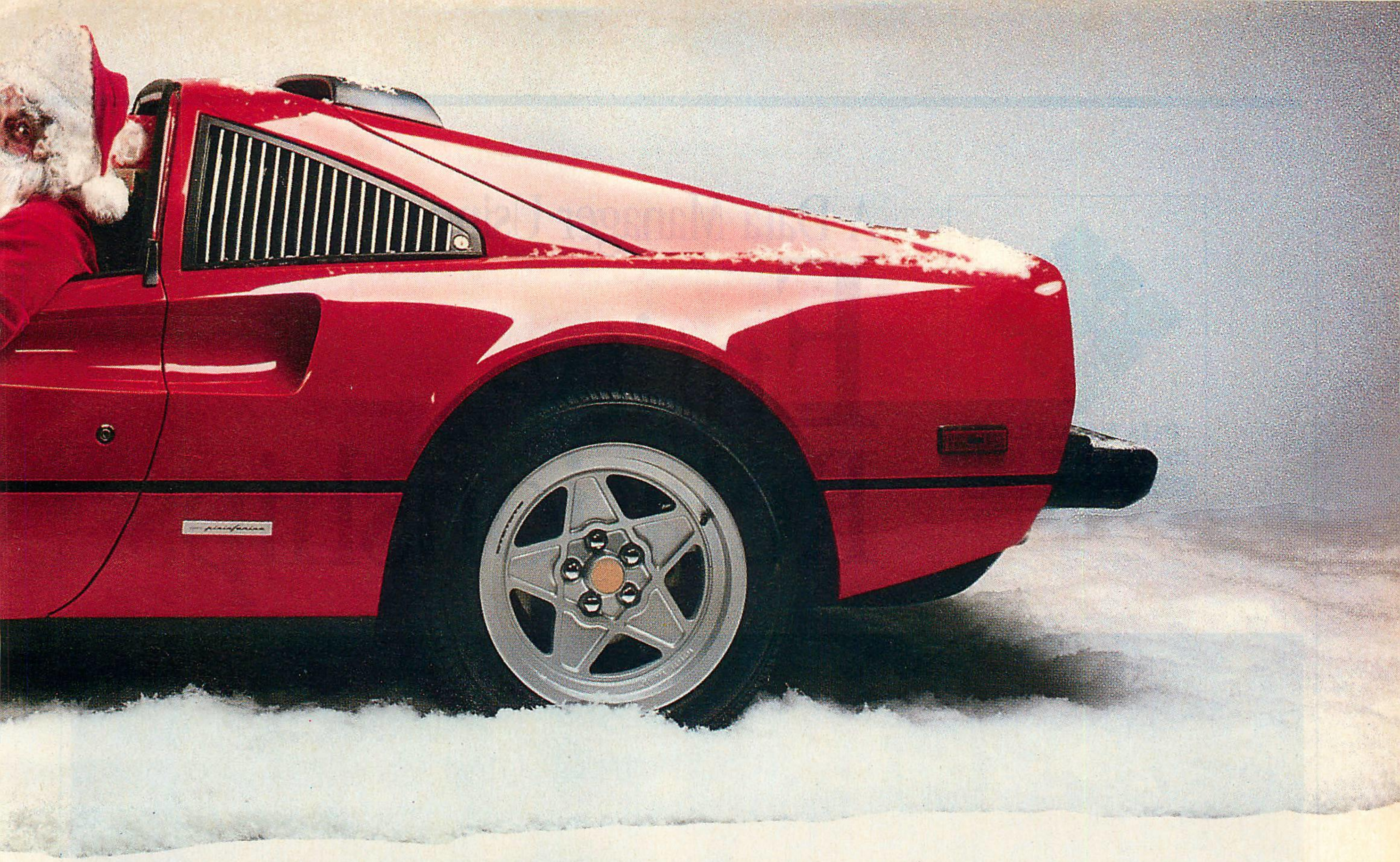
...Pmaker,™ the perfect program development manager...

Like UNIX™ Make and MMF, Pmaker updates and tracks modifications in your programs. With Pmaker, generating a final copy of your program is never more complicated than typing PMAKER MYFILE, regardless of the number of modules that make up MYFILE, their dependencies on one another, the number of modules changed, or the variety of compile/assemble/link steps required to generate the program.

...Pfix™86 Plus, the most advanced symbolic debugger on the market today...

With its adjustable window display of source file, disassembled object, data area, stack, breakpoint settings, CPU and coprocessor registers, Pfix takes the guesswork from debugging. You can debug without a listing, since you can see and enter symbolic names or absolute addresses in breakpoints, data displays, expressions, or with the in-line assembler. The entire 80xx and 80xxx series of

*Price effective through December 31, 1985.



y Pac™ For The Holidays.

5. $995.*

16-bit processors is supported, along with the user-assignable string and numeric variables, up to 100-step trace-back, and debug log to disk or printer. Breakpoints can be set in the source file window, whose display can be synchronized with the disassembled object during tracing; symbols can be added incrementally during the debug session; and the disassembly written to disk.

...Pfinish,™ the program that maximizes your program's efficiency...

Pfinish helps you turn your beta-test product into a software work of art. It analyzes your program or the entire operating environment during execution, and produces reports which tell you how much time was spent in each routine or interrupt, who called it, how many times, and much more. Wasteful, inefficient, or non-optimal areas of code become immediately apparent, whether they're in your code, the compiler library, or in the operating system routines themselves.

...Ptel™ gets the lead out of binary file transfer...

You get error-free file transfer and access to mainframes, minis, and micros. ASCII. XModem. Modem7. Telink and Kermit. MS-DOS® 2.x or 3.x. With Ptel. You can transfer 8-bit binary files over a 7-bit data path with Kermit. Masked file-name transfers with XModem. Or, transfer whole subdirectories with a single command using Telink. Ptel keeps track of the original file size and creation date. Ptel also offers a script language, backward scrolling, and the ability to handle DOS commands from inside the program.

CIRCLE NO. 205 ON READER SERVICE CARD

...And, Phoenix's new on-line software update service. Free to all registered Pfantasy Pac1 owners.

Pfantasy Pac1 software works in MS-DOS/PC DOS environments, is completely language-independent, and is available for the IBM® PC, XT,™ AT,™ and compatibles, the Wang® PC, the TI Business-Pro,™ and the Tandy® 2000.

So, spoil yourself a little. Get Pfantasy Pac1, and make your favorite programmer's dream come true.

For more information contact: Phoenix Computer Products Corporation, 320 Norwood Park South, Norwood, MA 02062. Or call: (800) 344-7200. In Massachusetts (617) 762-5030.

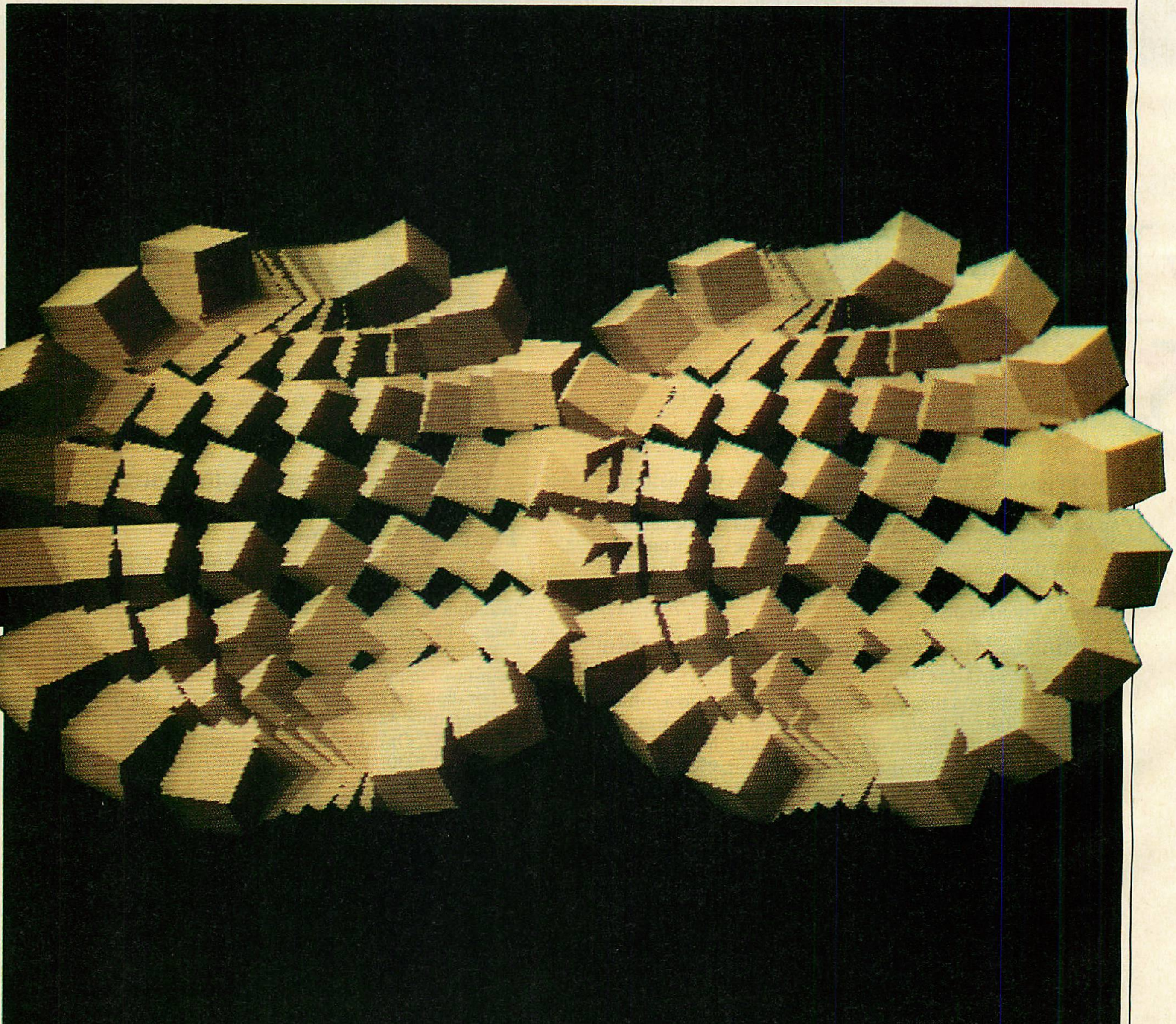
Programmers' Pfantasies™ by

Phoenix

SEE US AT FALL COMDEX

DATA MANAGERS AS DEVELOPMENT TOOLS

A Data Manager Using Entity-Relationships



COMPUTER GRAPHIC • DOV JACOBSON

ZIM implements an advanced data model in its applications development system that could make converts out of many system designers.

RICHARD M. FOARD

In 1978 a group of software developers began exploring the entity-relationship data model while writing programs for internal use by Bell Northern Research, Ltd., the Canadian counterpart of Bell Laboratories. Their work found widespread distribution and acceptance within Bell and eventually motivated the formation of Zanthe Information, Inc. Zanthe's stated goal was to create a fourth-generation applications development system for microcomputers that surpassed the capabilities of mainframe packages while providing the most powerful data management technology available.

Zanthe created ZIM, an applications development system built atop a sophisticated set of data management facilities. ZIM implements the entity-relationship data model, an extension of the ubiquitous relational model. Surrounding ZIM's data management facilities are a data dictionary, an interactive command and query processor, a programming language, a report generator, and a forms management facility for full-screen data entry and editing.

Although its query language, forms manager, and report generator make it suitable for end users, ZIM is a system developed and marketed by professional systems builders with a careful eye toward the requirements of applications developers. A basic end user/system developer package is available for the PC-DOS environment for $795. A runtime package tailored for use by value added resellers is available in quantities of 100 for $80 per copy.

Zanthe offers ZIM in several hardware/sofware environments in both single- and multiuser versions. Single-user ZIM runs on IBM PCs, PC/XTs, and PC/ATs under DOS 2.0 (or later) and configured with a minimum of two floppy-disk drives and 384KB of memory. Multiuser ZIM runs on ATs under IBM XENIX, on PCs, XTs, and ATs under Quantum Software's QNX operating system, and on the NCR Tower under UNIX System III or V.

ZIM can accommodate very large databases, consisting of as many as 9,800 files, 13 of which may be open simultaneously for processing by a single command. Any number of files, up to the maximum of 9,800, may be used during a single ZIM session. A single file may grow as large as 256MB. Field and record sizes are limited to the basic disk page size, a parameter the user specifies at database creation time to equal any multiple of 1,024 (bytes) between 1,024 and 8,192.

In a particular ZIM database, the limits on the number of records in a file or the number of fields in a record depend on the sizes of records and fields and on the choice of disk page size. A file of 100-byte records in a database with 1,024-byte disk pages is limited to 2,621,440 records, for example. Although, in theory, the number of fields in a record is limited to 8,192 by the maximum disk page size (assuming 1-byte fields), in practice it is not limited, because ZIM files are easily linked with one another using the *relationship* mechanism. Using relationships, numerous vertical slice files could be assembled into a single logical file with a monstrous number of data fields.

ENTITY-RELATIONSHIP

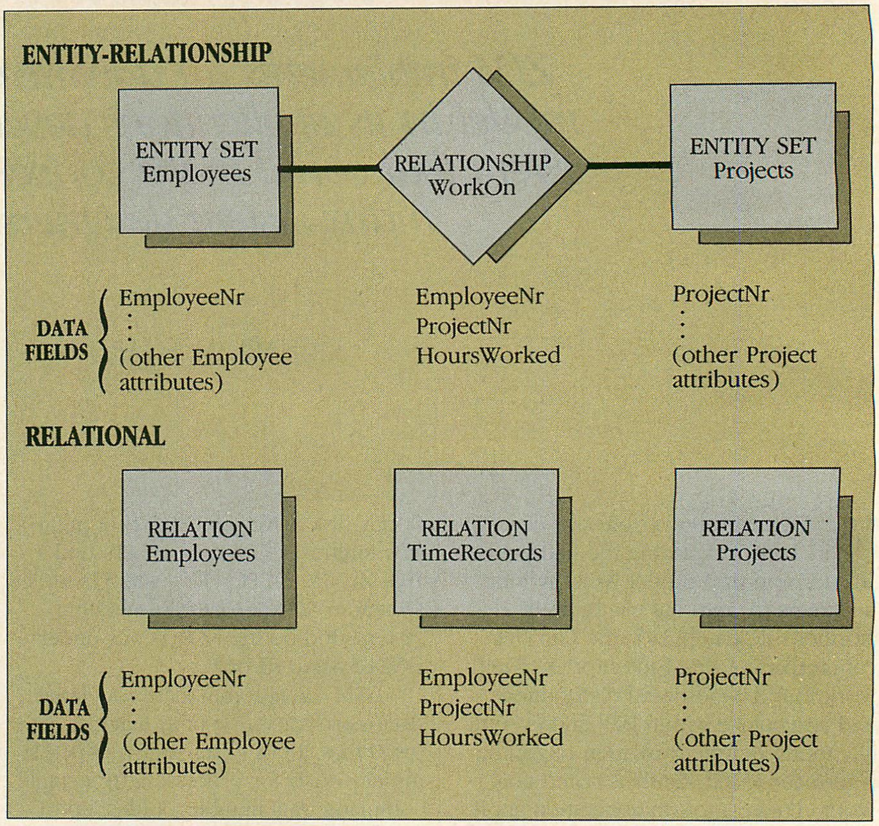
At the heart of every data management package lies its data model, a set of rules and formalisms providing the conceptual framework in which the applications developer envisions the real-world entities described in his database and in which he designs the structure of the database itself. ZIM's entity-relationship model is an outgrowth of the more commonly used relational model.

Both of these models represent collections of information about "things" in the real world, such as employees in a company or accounts in an accounting system, as simple tables of data. Most tables contain a key and a set of attributes that identify and describe a feature in the application domain. Under the relational model, these tables are called *relations*; under the entity-relationship model they are *entity sets*.

The two models differ in their representation of relationships between entities, such as that between employees of a company and the departments in which they work. Under the relational model, such relationships are represented by the appearance of a common field in the tables involved in the relationship. An employee relation, for example, would include a DepartmentNr field containing for each employee a value that can be found in the DepartmentNr field of the department relation; the common field establishes a logical linkage between the two relations.

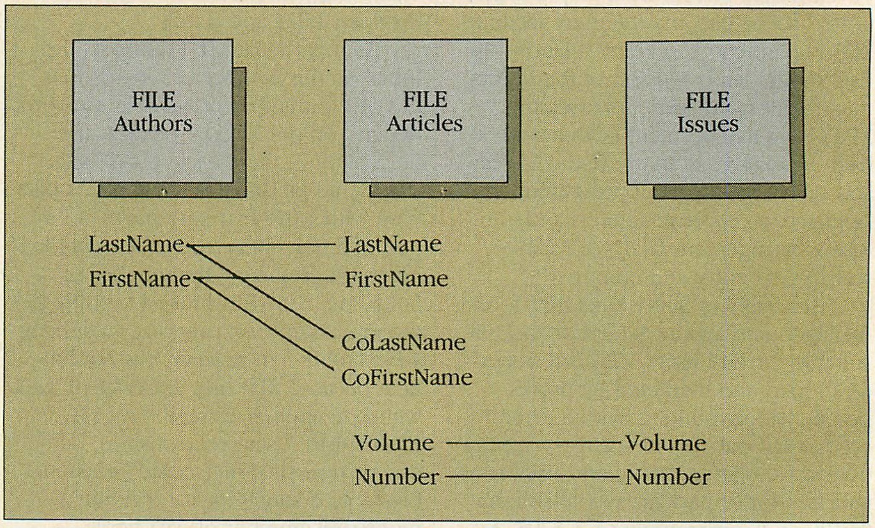
Under the entity-relationship model also, the DepartmentNr field would appear in both employee and department entity sets, but the relationship between the two sets would be coded explicitly in the database schema as a *relationship*, a construct with no counterpart in relational schemas. Relationships are given mnemonic names, such as WorkIn in the case of the employee-to-department relationship, and are formally specified in the database schema by the logical condition that defines them. The WorkIn relationship would be described by the condition Employees.DepartmentNr = Departments.DepartmentNr.

FIGURE 1: *Entity-Relationship Versus Relational Model*



The entity-relationship model is well-suited for representing many-to-many relationships such as the WorkOn relationship shown at the top of the figure above. The relational model of the same data is quite similar, but lacks the entity-relationship model's more complete representation of database semantics.

FIGURE 2: *Relational Model Database*



The two relationships in the sample database—one between authors and the articles they write or cowrite, and one between the articles and the issue in which they appear—are encoded by the appearance of common fields in the files.

All database management systems somehow must be able to handle complex, real-world relationships. The explicit recognition and formal encoding of relationships between entities is what distinguishes the entity-relationship model from the relational model. Where a relational system would, in effect, implement the relationship between employees and departments in its applications programs and in its user and design documents, an entity-relationship system implements it more directly—that is, at the deeper level of the database schema.

The ability to define named relationships between entity sets introduces a twist in the way database users can refer to collections of data. Referring to the data contained in an entity set is accomplished simply by naming the entity set, as in **list Employees**. A user refers to the data in *two* entity sets linked by a relationship by naming both entity sets with the name of the relationship in between, as in **list all Employees WorkIn Departments**. In ZIM terminology, both of these references to collections of data are called *set specifications*. Given the definition of WorkIn, the set of data specified by Employees WorkIn Departments is equal to the set a user of relational algebra would construct by performing a natural join of the Employees and Departments tables on the DepartmentNr field. ZIM sets are more general and shorter-lived structures than entity sets.

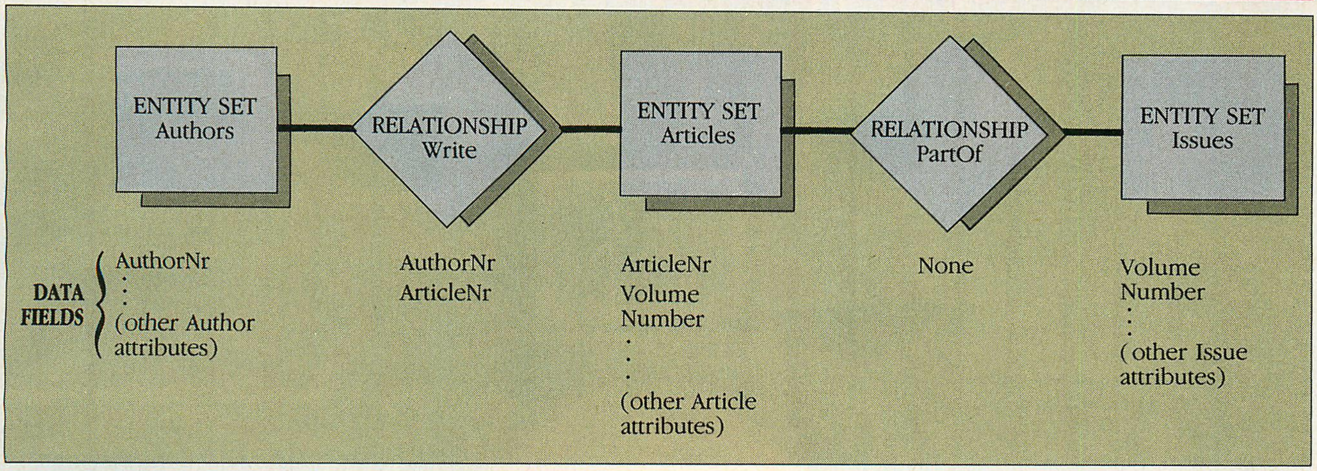
Relationships may carry data fields of their own just as entity sets do. In a database holding information about employees and the projects they work on, for example, a WorkOn relationship between employees and projects would parallel an employer's real-world requirement to keep track of which projects his employees are working on. The usefulness of maintaining information with relationships becomes especially clear in considering WorkOn and the task of maintaining time records. The amount of time that an employee spends on a particular project is a data item that does not fit naturally in either the employee's or the project's records. It is truly an attribute of both an employee *and* a project. Under the entity-relationship model, the problem is addressed by keeping time as a data field of the WorkOn relationship. The WorkOn relationship, then, would be described by the following condition:

Employee.EmployeeNr = WorkOn.EmployeeNr AND WorkOn.ProjectNr = Projects.ProjectNr

and it would be defined as having the three fields EmployeeNr, ProjectNr, and HoursWorked.

The WorkOn relationship, shown diagrammatically at the top of figure 1, is an instance of a many-to-many relationship. The entity-relationship model is particularly well-suited for representing such relationships clearly and logically. Interestingly, a diagram of a

FIGURE 3: *Entity-Relationship Model Database*



When the sample database was redesigned using the entity-relationship model, it was then possible to portray the many-to-many relationship between articles and authors that is sometimes found in the real world.

relational database holding the same information (shown at the bottom of figure 1) is quite similar, but lacks the connecting lines of the entity-relationship diagram. The presence of the connecting lines reflects the entity-relationship model's more complete representation of database semantics.

Both WorkIn and WorkOn are relationships between different entity sets. The model also supports relationships between members of the same entity set, such as the one-to-many Manage relationship between some employees and other employees. These relationships, termed *reflexive*, are defined in the same way as interset relationships.

ZIM allows synonyms, called *roles*, to be created for entity set and relationship names. Using roles, the database designer can create data specifications that read more naturally than they might otherwise. By creating the role Managers for the entity set Employees, for example, the designer can allow users to refer to Managers Manage Employees instead of the more obscure Employees Manage Employees.

Roles also can be used to make set specifications involving relationships read more naturally in context. Because the Manage relationship has no directionality as far as ZIM is concerned, the specifications Managers Manage Employees and Employees Manage Managers are equally valid and identify the same set of data. The meaning of the latter specification, however, is misleading to the user. ZIM's role mechanism allows the database designer to create a synonym for Manage, named WorkFor, so that users can think in terms of the more English-like specification, Employees WorkFor Managers.

THE SAMPLE APPLICATION

The sample database developed by *PC Tech Journal* for testing data management products contains three files holding information about magazine issues, articles, and the authors of these articles. (For a complete explanation of the sample application, see "Sample Application Specifications," August 1985, p. 48. The article also is available for downloading on PCTECHline.) Implicitly, it contains two important relationships, one between authors and the articles they write and another between articles and the issues in which they are scheduled to appear. As the sample database is structured, these relationships are encoded by the appearance of common fields in the files (figure 2).

Although the database could be imported into ZIM as originally defined, with three entity sets named Issues, Articles, and Authors, this reviewer chose to restructure it to bring it more in line with the entity-relationship data model shown in figure 3. The restructuring allowed the association of more than two authors with an article, whereas the original database, which carries author name and coauthor name fields in its Articles file, encodes only a one- or two-to-many relationship between authors and articles; in practice, the relationship can be many-to-many.

The first item of business in the restructuring was to devise a more compact key for identifying authors than that made up of their combined first and last names (this change was not dictated by the conversion to the entity-relationship framework, but it seemed worthy from a space management standpoint). The Social Security Number field was chosen for this purpose and was renamed AuthorNr. The author and coauthor name fields were removed from the Articles file and a numeric ArticleNr field was added for use as an identifying key. With these changes in place, the relationship Write was introduced, with data fields AuthorNr and ArticleNr. The defining condition for Write is:

```
Authors.AuthorNr = Write.AuthorNr AND
    Write.ArticleNr = Articles.ArticleNr.
```

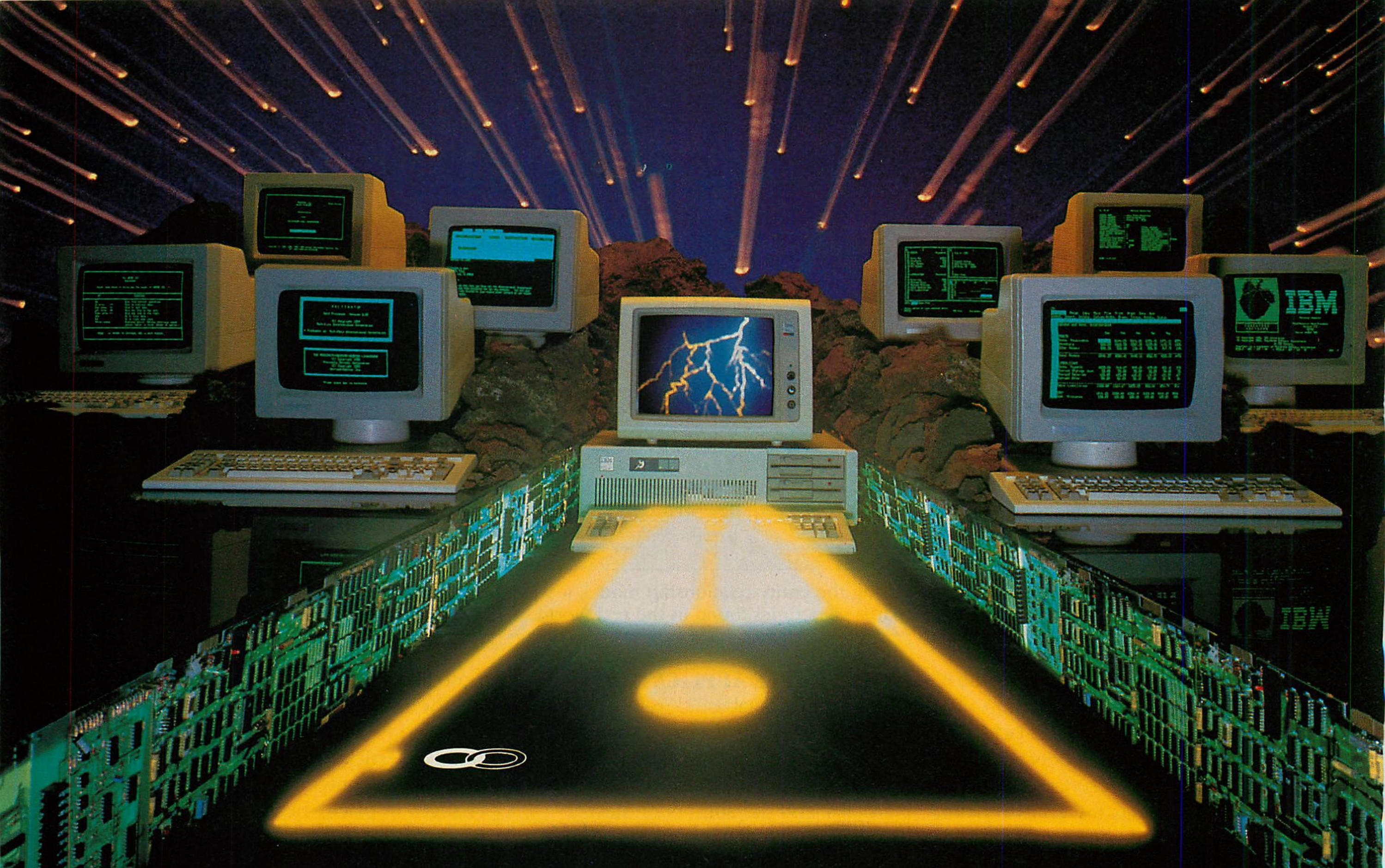
The many-to-one relationship betwen articles and issues was represented as the relationship PartOf, which has no data fields and which is defined by the condition:

Articles.Volume = Issues.Volume *and*
Articles.Number = Issues.Number

Finally, to avoid the awkward appearance of Articles Write Authors or Issues PartOf Articles, two roles, WrittenBy and Contain, were introduced to be synonymous with but reversing the implied directions of the relationships Write and PartOf. The ZIM data dictionary entries for the restructured database are shown in figure 4.

The data rearrangements necessary to produce the structure changes described above were accomplished using ZIM, by loading the three original data files straight into entity sets, then continuing, still using ZIM's facilities, in a stepwise restructuring procedure.

ZIM presents its user with a view of the database in which information resides in entity sets, relationships, and documents. These named collections of data may be arranged in a tree-structured set of ZIM directories, in the same way that DOS files are stored in the tree-structured DOS file system. ZIM



Networking Raised to a Greater Power

Advanced Technology. With it, IBM tripled the speed of the PC and increased its memory capacity five-fold. Nowhere is this increase in computing power more important than in networking situations. If the AT's technological advances have prompted you to look into a multi-user network, you owe it to yourself to take a closer look at MultiLink Advanced™ . . . a unique multi-tasking, multi-user networking system that runs programs under PC-DOS 3.0.

Eight Workstations for the Price of an AT. MultiLink Advanced™ represents the next generation in networking systems for IBM microcomputers. The system enables terminals, connected to a single AT, to emulate IBM-PC's having up to 448K of RAM (The PC-Shadow™ terminal, shown above, even has a PC look-alike, as well as work-alike keyboard and display).

This means that instead of spending $3,000 per workstation for a PC with a Kilobuck "Network Interface Board," you can use inexpensive terminals . . . eight of which cost less than an IBM AT. Even if you need only one workstation connected to your AT, you'll realize significant savings.

MultiLink Advanced™ . . . Instant Access to All of Your Resources. Central to most multi-user situations is the need to coordinate a variety of printers. With what's been described by *PC-Tech Journal* as ". . . by far, the best print spooler for the IBM PC," MultiLink Advanced™ gives users the option to print either at their workstations, or at a central location. In addition, programs and files can be shared by multiple users locally or through use of a modem. Just think of it . . . having remote access to an AT with a lightweight terminal/modem.

Although designed to take advantage of the AT, MultiLink Advanced™ runs on all versions of PC-DOS, except 1.0, and certain implementations of MS-DOS. A wide range of leading programs are supported which include WordStar, dBASE III, Multimate, and Lotus 1-2-3.

Get the Advanced Story Today. Call The Software Link Today for complete details and the dealer nearest you. MultiLink Advanced™ is immediately available at the suggested retail price of $495 and comes with a money-back guarantee. VISA, MC, AMEX accepted.

MultiLink™
ADVANCED

THE SOFTWARE LINK, INC.

8601 Dunwoody Place, Suite 632, Atlanta, GA 30338 Telex 4996147 SWLINK

CALL: 404/998-0700

Dealer Inquiries Invited

CIRCLE 143 ON READER SERVICE CARD

IBM PC, AT, & PC-DOS are trademarks of IBM Corp. MS-DOS, WordStar, dBase III, Lotus 1-2-3, and Multimate are trademarks of Microsoft Corp., MicroPro, Ashton-Tate, Lotus Development Corp., & Multimate International, respectively.

THE SOFTWARE LINK, INC./CANADA
400 Esna Park Drive, Suite 18
Toronto (Markham), Ont. L3R 3K2
CALL: 416/477-5480

MultiLink Advanced™ & PC-Shadow™ are trademarks of The Software Link, Inc.

creates a DOS directory for each ZIM directory that is defined. In fact, the mapping of all ZIM data structures onto the DOS file system is simple and direct: each entity set and document is stored in a single DOS file. Any indexes associated with an entity set physically reside in the same DOS file that contains the set's data. Each relationship that contains data is also stored in a single DOS file (those without data need no storage other than that required in the data dictionary to represent their defining conditions).

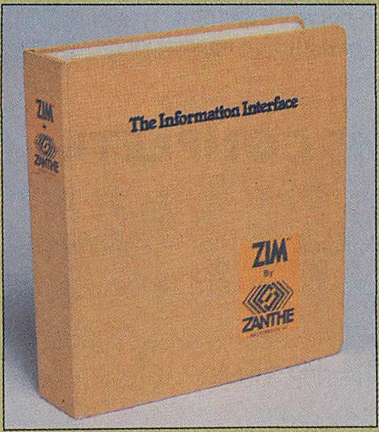
ZIM documents are bridges between ZIM and the DOS file system. They provide a way for users to store and manipulate unstructured text files under ZIM. By creating a document and associating its name with the name of an existing DOS text file, for example,

ZIM OVERVIEW

ZIM 2.4

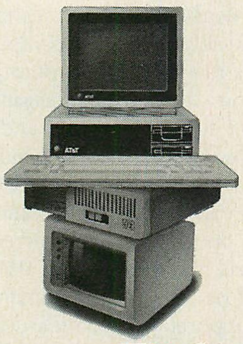
Zanthe Information, Inc., 1785 Woodward Drive, Ottawa, Ontario K2C 0R1. 613/727-1397.

Product type. Database management and fourth generation applications development system for business use.
Software Environment. MS-DOS, PC-DOS 2.x and higher, Double DOS, UNIX, XENIX, and QNX.
Network support. LANs supported in single-user mode only.
Hardware environment. IBM PC, PC/XT, PC/AT, IBM compatibles, and multiuser system with a minimum of 384KB of RAM and two floppy-disk drives. Additional RAM, RAM drive, system-compatible hard-disk drive, and 8087 coprocessor can be supported.
User interface. Program is command language and menu-driven, and uses macros/procedures and function/control keys. The command language, data definition, and data manipulation are all English-like.
File limitations. 8,192 characters per field; unlimited characters per record; unlimited fields per record (possible when entity sets are linked together); and 2 million records per file. The number of open files is limited only by the operating system. There are unlimited indexes per file, and the program allows a file to span multiple disk-volumes (using the UNIX and XENIX versions only). Data are stored in variable length records. The basic architecture is entity-relationship.
Access to system facilities. From within the program the user can access all operating system functions.
Modification facilities. The program can merge two or more files into a single file, and can split a file into two or more files. The program can update a file with data from another file and can update multiple files simultaneously.
Help facilities. A written tutorial is provided with the program.
File design. The product uses screen-painting method of entry-screen design allowing custom data entry screens. The user may create multiple



data entry screens for a single file. The program supports derived fields with the information derived as a result of calculations from another file as well as from a user-supplied list or file of acceptable values. Other field attributes allow for view-only fields, numeric fields, and must-enter fields.
Data entry. The program automatically checks for duplicate entries in a file; provides range checking functions; allows the user to supply standard entry values during entry; and provides facilities for batch data entry.
Query and sorting. Search facilities allow for partial key search, selection operators, and logical operators. Sorts may be performed in ascending and descending order. The product supports multiple sort operations and multiple indexes on unlimited fields. Query and ordering specifications may be saved for repeated use. Automatic updating of indexes is provided.
Reporting. The program produces label reports. The report formats can be saved and edited and can contain information from two or more files. The program also produces summary reports. These reports can include totals and subtotals, control breaks for pagination, calculated results, parenthetical control of order of operations, averages, statistical, and other math functions. Final reports can include headings/footers and pagination. Reports can be directed to the screen, printer, or choice of multiple printers, a disk file, or an import export file format. Print enhancements may be used and paper size, margins, etc. specified in the report definition.
Security. Programmable.
Utilities. File maintenance and back-up utilities are provided. Print-outs of report, file-design definitions, and automatic optimal data access strategy selection are also supported.
Applications development facilities. Customization possible with macros/procedures and custom menu generation. Program provides fully programmable procedural language, links to DOS applications or other languages, and generates turnkey applications. Runtime compiler or module is available for applications developed.
Data compatibility. The program is able to read and write comma-delimited ASCII, fixed length ASCII, and any delimiter file formats.
Distribution. First delivery: December 1983; current version first delivered: January 1985; number of installations: 700. Primary method of distribution is through dealers and direct sales.
Price. Retail: $795; multiuser UNIX version: $1,800; multiuser XENIX version: $2,500; compiler: $125.
Support. The product includes on-screen tutorials, sample applications, telephone support, demonstration disk, and update/maintenance plan. Updates are available for $119.25. The product is copy-protected, but backups are permitted for personal use. Runtime licenses are available in quantities of 100 for $80 each.

This material is taken from Data Decisions Microcomputers, *a three-volume monthly updated reference service that evaluates microcomputer systems, software, and peripheral products. For additional information contact Data Decisions, 20 Brace Road, Cherry Hill, NJ 08034. 609/429-7100*



the ZIM user makes the text file available for listing to the terminal, for providing imported data, or for receiving exported data. ZIM can be instructed to direct its output to a document instead of to the terminal, allowing the user to produce reports and other text output in DOS files that may then be processed outside ZIM by print spoolers, text editors, spreadsheet programs, or other facilities. Because user-composed programs reside in documents, users can prepare programs outside ZIM using their favorite text editors, then make the program files known in ZIM by associating document names with the names of the program files.

A DOS DIR listing of a directory containing a ZIM database reveals a set of meaningless but regularly structured file names such as ZIM0301 along with any mnemonic names the user has associated with various document files. The regular structure of file names invented by ZIM makes for convenient en masse back-up, copying, or deletion of ZIM databases using DOS commands, such as COPY ZIM*.* A:. The ZIM release package includes a utility program called Zimfiles that can be invoked from DOS to produce a listing of each object in a ZIM database along with the name of its corresponding DOS file.

ZIM DATA TYPES

ZIM data fields and variables may be defined as having one of seven data types. Character string data are stored in fields of type Char or VarChar. Char fields are fixed in length. VarChar fields are allowed to vary in length up to a specified maximum size. Database designers can conserve disk space by typing as VarChar large text fields that frequently are not filled.

Numbers of up to 15 digits, possibly including fractional parts, can be represented in ZIM's Numeric data type. The less general, more compact Int, LongInt, and VastInt data types are available for representing signed, integer-valued data and carry 16, 32, and 64 bits of precision, respectively.

ZIM supports a Date data type that stores dates in numeric, YYYYMMDD form. Arithmetic can be performed on data items of type Date; numbers of days may be added to or subtracted from dates, giving other valid dates.

Fields or variables of any type may take the special value $null, indicating unknown. Unknown values are handled consistently throughout ZIM, in a way that can help developers find bugs and oversights in an application that result from uninitialized variables. Any arith-

FIGURE 4: *Data Dictionary*

```
Entity Sets
  Articles
  Authors
  Issues

Fields
  SN FieldName      OwnerName     Type      Length Decimals Reqd Index

  (Articles Entity Set)
    1 ArticleNr     Articles      Int          5       0    yes   yes
    2 Volume        Articles      Int          1       0    yes   yes
    3 Number        Articles      Int          2       0    yes   yes
    4 Category      Articles      Char        17       0    no    no
    5 Department    Articles      Char        21       0    no    no
    6 Title         Articles      Char        60       0    no    no
   11 Commissioned  Articles      Char         1       0    no    no
   12 DateRecd      Articles      Date         8       0    no    no
   13 EditPages     Articles      Int          2       0    no    no
   14 ListPages     Articles      Int          2       0    no    no
   15 Payment       Articles      LongInt      4       0    no    no
   16 Bonus         Articles      LongInt      2       0    no    no

  (Authors Entity Set)
    1 LastName      Authors       Char        18       0    yes   no
    2 FirstName     Authors       Char        12       0    yes   no
    3 Address       Authors       Char        20       0    no    no
    4 City          Authors       Char        16       0    no    no
    5 State         Authors       Char         2       0    no    no
    6 Zip           Authors       Numeric      5       0    no    no
    7 WorkPhone     Authors       Numeric     10       0    no    no
    8 HomePhone     Authors       Numeric     10       0    no    no
    9 AuthorNr      Authors       Numeric      9       0    yes   yes
   10 Biography     Authors       VarChar    200       0    no    no

  (Issues Entity Set)
    1 Volume        Issues        Int          1       0    yes   yes
    2 Number        Issues        Int          2       0    yes   yes
    3 Deadline      Issues        Date         8       0    no    no
    4 Month         Issues        Char         9       0    no    no
    5 Year          Issues        Char         4       0    no    no

  (Write Relationship)
    1 AuthorNr      Write         Numeric      9       0    yes   yes
    2 ArticleNr     Write         Int          5       0    yes   yes

Relationships
  RelName        RelCondition
  Write          Authors.AuthorNr=Write.AuthorNr
                           and Write.ArticleNr=Articles.ArticleNr

  PartOf         Articles.Volume=Issues.Volume
                           and Articles.Number=Issues.Number

Roles
  RoleName       OwnerName
  WrittenBy      Write
  Contain        PartOf
```

The data dictionary is composed of eight predefined entity sets. The database structure is stored in the four special entity sets: EntitySets, Fields, Relationships, and Roles; Forms and Formfields hold the descriptions for data entry forms; Documents holds the correspondence between ZIM documents and DOS file names; and Variables stores the names and types of ZIM variables.

metic expression involving $null evaluates to $null; any comparison of any value with $null evaluates to false.

Like many other data management systems, ZIM uses its own facilities to encode the structure of a database. The structure description of a ZIM database is itself a ZIM database.

Eight predefined entity sets exist in every ZIM database and, together, make up its data dictionary. The structure of entity sets and relationships is stored in four special entity sets: EntitySets, Fields, Relationships, and Roles. Data entry-form descriptions reside in the entity sets Forms and FormFields. The Documents entity set encodes the correspondence between ZIM document names and DOS file names, and the Variables entity set holds the names and types of all defined ZIM variables.

A simple ZIM database can be created by first making entries in the Fields entity set that give the name and type of each field in the database and the name of its *owner* entity set, then making an entry in the EntitySets entity set for each entity set (see figure 4).

Once the required descriptive information has been added to the data dictionary entity sets, the ZIM **create** command can be used to make the new objects available for use. **Create** serves a function analogous to a compiler's function in program development. Creating an entity set whose field descriptions have been entered in the Fields entity set, for example, "compiles" the field definitions into a permanent internal representation of the structure of the entity set that can be revised only by using the **erase** command to discard the definition, then using the **create** command again.

Because ZIM uses itself to encode database structure, the process of defining a new database structure in ZIM is one of pure data entry, exactly as the initial loading of an application data file would be. The simplest (and most shortsighted) way to define a database is to use ZIM's **add** command interactively and key in the descriptions of all fields, entity sets, and so on, directly from the terminal. Because ZIM's provisions for changing the structure of a database once it has been defined are limited, a wiser approach is to use a text editor to create columnar text files containing database structure information, then use ZIM's ability to import data into an entity set from a delimited text file to add data dictionary information.

The process of defining a new database structure in ZIM is one of pure data entry, exactly as the initial loading of an application data file would be.

Assuming that a user had created columnar text files named Fields and Entsets, describing the fields and entity set of a database, he could then create a ZIM database by giving the following commands:

```
add documents let docname = fielddoc
  filename = fields
add documents let docname = setdoc
  filename = entsets
create document fielddoc in zim
create document setdoc in zim
add Fields from fielddoc
add EntitySets from setdoc
```

The first pairs of **add** and **create** commands turn the user's text files into ZIM documents. The second pair of **add** commands performs a bulk load of the Fields and EntitySets components of the data dictionary. The definition process can then be completed by executing a **create** command for each of the entity sets in the database.

The advantage of bootstrapping database structure information in a batch from externally created text files is that the database can be quickly reconfigured by editing the text files, reinitializing the database directory, and then repeating the generation process. In fact, the creation commands themselves can be stored in a ZIM program document and executed by typing in the name of the document, thereby reducing the whole regeneration process to only a few keystrokes.

Users timid about tackling the task of defining a ZIM database for the first time may wish to take advantage of the Interactive Definition Facility (IDF). The IDF, itself a ZIM application, leads the user through the steps necessary to define ZIM objects using a menu-driven process. The IDF also serves a tutorial function by showing the direct ZIM commands it is issuing in order to accomplish the definition work the user has commissioned.

In a typical IDF session, the user might define an entity set, then ask IDF to generate a form for use in updating the entity set as well as a short ZIM program that can be used to load the form and then to accept data entry.

Although IDF can be used to construct a very simple application for displaying and updating the contents of a database, it is mainly intended as a teaching and familiarization tool. It cannot be used to build a complex application. Zanthe's documentation gently but pointedly suggests that users learn to use ZIM directly, at its command level, as quickly as they can so that they can eventually leave IDF behind.

The entire program text of the IDF application is distributed with ZIM. Because IDF itself is a substantial ZIM application, it is useful even to sophisticated first-time users as a model for applications building in ZIM.

ZIM FEATURES

Any field in a ZIM entity set or relationship may be indexed. ZIM indexes are built in B-tree form and can be applied both to fields containing unique values and to those with multiple occurrences of values. The database designer must specify whether or not a field is to be indexed at the time of its definition by setting an attribute in the Fields entity set—before issuing a **create** command for its owning relationship or entity set. Indexes may be attached to single fields only; ZIM does not support indexes on compositions of several fields.

Index integrity is maintained transparently across all database updates. Once attached to a field, an index may not be removed. ZIM also does not support after-the-fact indexing. Because the physical structure of the disk files in which entity sets and their associated indexes are stored is determined by the data types of component fields and indexes, entity sets must be unloaded, redefined, and reloaded if the user wishes to add an index.

ZIM's medium of data interchange with the outside world is the delimited, line-oriented ASCII text file. Data may be imported to an entity set from a columnar text file by directing ZIM's **add** command to take its input from a document instead of from the terminal. Data fields in the input document may be delimited by a user-specified character,

such as comma or space. The user need only make the text file known to ZIM by creating it as a ZIM document and specify the delimiting character using a **set delimiter** command.

Input data conversion rules are natural. Numbers may appear in the input text file with or without decimal points, regardless of whether they are bound for fields of integer types (Int, LongInt, or VastInt) or fields of the more general Numeric type, in which decimal places may be represented. Input numbers containing more precision than can be represented in their destination fields are automatically rounded. Character strings exceeding the maximum widths of their destination Char or VarChar fields are truncated.

The only ZIM data type that is likely to require some reformatting work when importing data from plain text files is the Date type. ZIM accepts dates only in the eight-digit, YYYYMMDD form, a common internal representation in data management circles, but one not likely to be found in text files produced by other programs, especially those that produce text files intended to be read by people.

Exporting data from ZIM is also accomplished by using a command normally meant for interacting through the terminal with the user. The ZIM **list** command, used in conjunction with ZIM's capability to redirect terminal output to a document file, can be used to create a columnar text file containing data from entity sets, relationships, or user-specified combinations of entity sets. Unfortunately, ZIM's facility for exporting data in this way is not as flexible as its facility for importing data in that there is no convenient way to select a special delimiter character to separate outbound data, as there is for inbound data. To separate outgoing fields with commas instead of spaces, for example, the user must accompany the **list** command with a format specification naming every field of the data being unloaded and explicitly including each separating comma.

Although it makes no restrictions on adding new entity sets or new relationships to an existing database, ZIM does not provide for the dynamic restructuring of previously established elements of a database. As mentioned earlier, adding an index to an entity set requires unloading its data, changing the definitions of its component fields in the data dictionary, then reloading the data. The same sequence of operations is necessary to change the data type of a field or to add or remove fields.

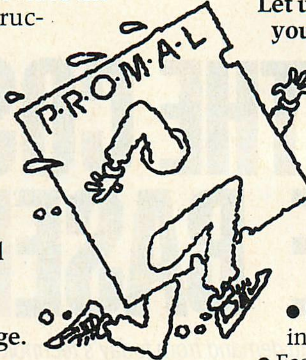
Although ZIM has no built-in support for restructuring, its command set and data dictionary facilities provide a means for constructing automatic procedures that make the unload/reload process of restructuring fairly painless. In fact, Zanthe includes a pair of ZIM programs for unloading and reloading arbitrary entity sets with the current release of the product. These programs, **unloadent** and **reloadent**, can be used without modification for simple restruc-

Adding an index to an entity set requires unloading its data, changing the definitions of its component fields in the data dictionary, then reloading the data.

Now there is an even better structured, compiled programming environment than PROMAL. Introducing PROMAL 2.0 for the IBM PC, the Apple II, and the Commodore 64.

Until now, the best next language for the serious programmer was PROMAL™ Now, it's the new PROMAL—PROMAL 2.0.

PROMAL 2.0 provides the same sophisticated structured programming environment, the same fast one-pass compiler, the same speed of execution, the same powerful commands of the earlier release —plus a host of useful new features.



Not just a language. A complete programming environment.

PROMAL—the PROgrammer's Micro Application Language—provides you with a complete programming environment, including a structured, high level language, a powerful program Editor, and a compiler that quickly turns your source code into compact, rapidly executing object code. Plus a library of integrated machine-language subroutines for frequently used tasks. And for the Apple II and the Commodore 64, PROMAL provides a DOS-like operating system Executive.

PROMAL 2.0— Even more of a good thing.

In addition to all of the features that have made PROMAL users declare it "the best language I've ever used," PROMAL 2.0 provides:

- Overlays that can be compiled separately for modular programming.
- Program size greater than 64k. (IBM PC only).
- True machine-to-machine portability.
- True 808X object code for the IBM PC.

Let us prove that PROMAL is your best next language!

Buy PROMAL 2.0 and try it for 15 days. If you don't believe it's your best next language, just return it for a full refund.

PROMAL Features

- Structured language with indentation.
- Fast, one-pass compiler.
- Simplified syntax.
- No line numbers.
- Multi-dimensional arrays, strings and pointers.
- Long variable names.
- Global, Local variables.
- Byte, Word, Integer & Real types.
- Decimal or Hex numbers.
- Functions and procedures with passed arguments.
- Built-in I/O library.
- Control Statements: IF-ELSE, IF, WHILE, FOR, CHOOSE, BREAK, REPEAT, INCLUDE, etc.
- Compiler I/O from/to disk or memory.

Executive*

Command oriented with line editing.
Allows multiple user programs in memory at once.
Function key redefinition.
Program abort or pause.
22 resident system commands.
Unlimited user-defined commands.
Prior command recall/edit.
I/O redirection to disk or printer.
Batch jobs.

Editor

Full-screen, cursor driven.
Function key controlled.
Line insert, delete, search.
String search and replace.
Block copy/move/delete/read/write.
Auto indent, undent support.
Edit after error.

Library

50 machine language commands.
Memory resident.
Call by name with arguments.
Formatted real output, string operations and much more.

*Apple II and Commodore 64 only. Requires one disk drive and 80-column card for Apple (IIe, IIc only).

1-800-762-7874
In NC: 919-878-3600

Systems Management Associates
3325 Executive Drive, Dept. TJ–1
Raleigh, North Carolina 27609

SMA

NEW for PROMAL The Graphics Toolbox*

Twenty fast subroutines for creating sophisticated, high-resolution graphics, including windows, clipping, scaling, and text-on graphics using scaled, rotated, user-defined fonts. $29.95.

*Available for the Apple II and the Commodore 64

Order Form

My system is (check one)
☐ IBM PC/100% compatibles ☐ Apple IIc/IIe
☐ Commodore 64/128

☐ Developer's Version—Compiler, Editor, Library, Demo disk, 280-page manual, (plus Executive for Apple and C-64) and stand-alone program generation. **$99.95 + 5.00 s/h.**

☐ End-User System for Apple II and Commodore 64—all features of Developer's version except stand-alone program generation. **$49.95 + 5.00 s/h.**

☐ Demo System—32-page "Meet PROMAL" manual and demonstration disk. **$10.00 + 2.50 s/h.**

☐ Graphics Tool Box for PROMAL—Available for Apple and C-64 only. **$29.95 + 2.50 s/h.**

☐ My check is enclosed.
☐ Please charge to my
___ Visa ___ Mastercard

Card Number ______ Expiration Date ______
Signature ______
Name ______
Address ______
City, State, Zip ______

NC residents add 4-1/2% sales tax.
Foreign orders add $15.00 additional s/h.

CIRCLE NO. 180 ON READER SERVICE CARD

GP TOOLS™

REESTABLISHES THE COST TO PERFORMANCE BASELINE.

Allen-Harris Industries, Inc. brings you the price and performance that you demand from today's technology. With a unique data repository concept, GP-Tools breaks through the boredom of data base management with GP-File, GP-Print and GP-Utility.

GP-FILE

- Data file attributes and screen display formats generated in one step.
- Instantaneous screen displays.
- Data retrieval in a fraction of the time found in other systems... regardless of primary or secondary key assignment.
- Security with encrypted password protection for your data.
- Up to 99 sub-views of your master data file display (each with a password option).

GP-PRINT

- Design your reports on the screen... graphically!
- Produce an unlimited number of sophisticated reports for each data file.
- Wide variety of printers supported...plus, custom printer option!
- Send reports to screen, printer or disc.
- Modify report formats easily.

GP-UTILITY

- Modify data file attributes or screen display formats.
- Sort files with parameters to enhance reports.
- Obtain data file structure information.
- Change password for maximum protection.
- File management.

$69.95

ALLEN-HARRIS INDUSTRIES, INC.
130 UNIVERSITY PARK DRIVE - SUITE 185
WINTER PARK, FLORIDA 32792
TELEPHONE 305/657-1144

GP-TOOLS a division of Allen-Harris Industries, Inc.
GP-TOOLS includes GP-File, GP-Print and GP-Utility.

To Order Or For More Information:

Call 1-800/821-2165 • ***in Florida 305/657-1144 (collect)*** • ***Major Credit Cards Accepted***

CIRCLE NO. 135 ON READER SERVICE CARD

turing tasks, and provide a starting point and model for users facing more complex restructuring requirements.

INTERACTIVE LANGUAGE

ZIM's command language is English-like and has a fixed grammar. Its core commands are **list**, **add**, **change**, and **delete**. **List** displays data from the database, **add** accepts data to be added, **change** allows modification of existing data, and **delete** removes data. All commands take their input from and produce their output on the terminal by default, but can be directed to read from or write to ZIM documents or sets instead.

The command set is simple, deriving its simplicity by relying on the expressive power of the entity-relationship model and by exploiting ZIM's ability to accept a general set specification as an argument to any of its basic commands.

All ZIM's basic commands operate on *sets* of data (not to be confused with *entity sets*). Using ZIM's set-building abilities, a user can specify collections of data from entity sets, subsets of entity sets, relationships with data fields, combinations of data from entity sets related by relationships, aggregates of data from other sets, and so on, with brief, English-like phrases. In fact, nearly every data management command in ZIM takes a *set specification* as an argument. In the command **list Employees**, for example, Employees is a simple set specification that specifies all members of the entity set Employees. Set specifications may include logical expressions that select subsets of other sets, such as Employees where age > 45, or Departments where Billable = 'Y'. More complex set specifications may refer to several entity sets related by relationships, such as Employees WorkFor Managers WorkIn Departments.

A set specification may also include a *sorted by modifier*. These modifiers indicate that the set is to be assembled in sorted by combinations of fields in ascending or descending order. Expressions involving fields may also be given in *sorted by* specifications.

A set specification is the only context in which data can be sorted. Since sets survive only as long as a ZIM session, ZIM's sorting facilities cannot be used in the traditional sense of sorting a file. In fact, ZIM cannot permanently reorder an entity set or relationship except by creating a new copy from a sorted set. Although this may sound like a painful restriction, it really is not because permanent elements of the database are accessed via the set-building mechanism in every context in which a user might desire a sorted order, such as in producing sorted reports or obtaining answers to ad hoc queries.

Delete is the simplest of ZIM's core commands, allowing the removal of entries in entity sets or relationships with fields. The **add** command adds entries to entity sets or to relationships with fields, taking its input from sets, documents, or directly from the terminal. **Change** is similar in syntax to **add**, although its operation is not restricted to simple sets; **change** can be applied to fully general set specifications and, like **add**, can take input from sets, documents, or the terminal. Both **add** and **change** can be accompanied by a list of **let field = expression** phrases that restrict the commands' operations to a subset of the fields in the set being operated on (if an **add** command specifies the values of just some of the fields in a set, the unspecified values are by default given the value $null).

The **list** command provides a general query and simple reporting facility. It can be applied to any set or document, modified by *where* and *sorted by*

phrases, and accompanied by a format specification. Format phrases can contain field names, expressions involving fields and variables, and constant text, appearing in arbitrary order.

ZIM's **find** command accepts a general set specification and, following this specification, assembles a set. Unlike **list** and **change**, which discard sets after simultaneously assembling and operating on then, **find** builds sets that may be used in subsequent commands.

The user can assign names to the sets assembled by **find** or can leave them unnamed. Those sets left unnamed can be referenced implicitly by subsequent commands. The three command sequences are equivalent:

```
list Employees,
```

and

```
find Employees
list,
```

and,

```
find Employees —> EmpSet
list EmpSet
```

ZIM maintains a current position in each active set. Once a set has been assembled, ZIM's *set browsing* commands—**top**, **bottom**, **next** and **previous**—allow the user to move his current position around within the set. The **sort** command can also be applied to sets assembled by **find**.

Find and the family of set browsing commands allow users or user programs to factor a complex action or query into a sequence of simpler actions, as in the sequence:

```
find Employees WorkIn Departments where
  DepartmentNr = 10
sort by LastName
list all where Salary > 32000 format
  LastName Salary/52
```

PROGRAM DEVELOPMENT

ZIM includes a full programming language from which all its data management, forms management, and report generating facilities can be invoked. Two levels of programming support are provided. A command macro facility allows frequently used sequences of commands to be saved in document files and invoked by typing the name of the document. More demanding tasks can be accomplished by defining ZIM procedures, which provide local variables and more sophisticated parameter-passing and may be compiled for faster execution. A procedure also may be invoked interactively, by typing its name accompanied by a (possibly empty) list of parameters enclosed in parentheses.

The macro and procedure languages are almost identical. Macro parameter substitution is not available in procedures, and, conversely, procedures' formal parameter declaration constructs may not be used in macros.

Programs in either form are prepared in ordinary DOS text files using whatever text editing program the user favors, then made accessible from within ZIM via ZIM's document facility. The user need not leave ZIM to edit a program file; ZIM provides two ways of invoking DOS commands from the ZIM command interface. Using the **system** command, any DOS-executable program can be executed from ZIM, including a text editor; control is returned to ZIM when the command is finished. If the user can arrange to name his text editor EDITOR.COM or EDITOR.EXE, he can invoke it with the ZIM command **edit**. The advantage of using the latter method, aside from saving a few keystrokes, is that ZIM notices when the text of a compiled procedure file is edited and advises the user that his executable version will not be current after the edit. Sufficient memory, over and above ZIM's minimum requirement of 384KB, must be present in the machine to run a program invoked from within ZIM.

ZIM's programming language is modern and rational. The standard, block-structured IF..ELSE..ENDIF and WHILE..ENDWHILE control structures are available for use in both macros and procedures. Assignment statements are always introduced by the LET keyword (as is also the case in BASIC) and they take the familiar **variable = expression** form. Expressions in assignment statements and other contexts may include the standard complement of arithmetic operators and can refer to an extensive set of built-in string manipulation and mathematical functions, including trigonometric functions.

In addition to assignment statements, control statements, and procedure and macro calls, ZIM programs may include any of the commands in its directly executable user command set, including commands to build, manipulate, and browse through sets, commands to load forms and display or accept information through them, and commands to generate reports. A simple ZIM procedure to cull and report data from two related entity sets is shown in listing 1.

Although macros are less flexible than procedures, they can, nonetheless, be quickly and easily constructed. A macro may be passed as many as nine parameters from a command entered by an interactive user. Parameter values are referenced from within macros as #1 through #9. Procedures can be invoked from within macros, but macro parameter values may not be passed as parameters to procedures.

Formal parameters to procedures can be used more flexibly than macro

Global variables can be used in macros or procedures, and they may be declared with any of the basic data types that are available for data fields.

parameters. Procedure parameters may be designated **in**, **out**, or **inout**; **in** parameters are passed by value, **out** and **inout** parameters by reference. **In** parameters may be used as local variables within procedure bodies just as C's scalar formal parameters may be. System developers accustomed to standard general-purpose languages may feel a little inconvenienced because this is ZIM's only mechanism for declaring local variables in procedures.

Global variables can be used in macros or procedures and may be declared with any of the basic data types available for data fields. Their declarations do not appear in program files as a conventional language user might expect, however. Instead, each ZIM data dictionary contains a Variables entity set in which the name, type, and length of each global variable must be declared. Global variables are declared by adding an entry to this entity set. ZIM variables are quite global indeed—they continue to exist and hold their values throughout an entire ZIM session. This characteristic can be exploited by applications developers in order to maintain long-lived global information or to pass information from one program to another.

Notably absent from ZIM's programming language are arrays. Programmers faced with the requirement of storing lists of like-valued data items must resort to using sets or entity sets in the database even if their list storage requirements are local to a procedure.

Through a set of predefined variables, ZIM makes useful information about the environment accessible to

user programs, such as the current date and time ($Date and $Time) as well as the number of set members found by the most recently executed **find** command ($SetCount). Other predefined variables provide for convenient, transparent communications between user programs and the forms manager and report generator.

Both macros and procedures can be invoked and executed interpretively from the user command level or from other macros and procedures. By developing a set of procedures for interactive use, an applications developer can hide the more complex aspects of an application and package a friendly command set for a nontechnical community of database users. ZIM's interactive command processor is also useful to the developer in the process of coding and testing procedures. Its **parse** command performs a complete syntax check on commands or procedures without executing them, allowing programmers to do dry runs of procedures without risking damage to the database from a partially correct procedure.

ZIM also provides three levels of command tracing. Activation of the **flow** option causes each procedure or macro to announce its name when beginning or ending execution. If the **trace** option is turned on before or during interpretive command execution, each procedure or macro statement is shown at the terminal before it is executed. The **lextrace** option supports even finer-grained debugging; it shows each character of a command as it is parsed by the interpreter and gives the substituted values of macro parameters. All three of

Programs in either compiled or interpretive execution automatically use an 8087 or 80287 numeric coprocessor if one is present.

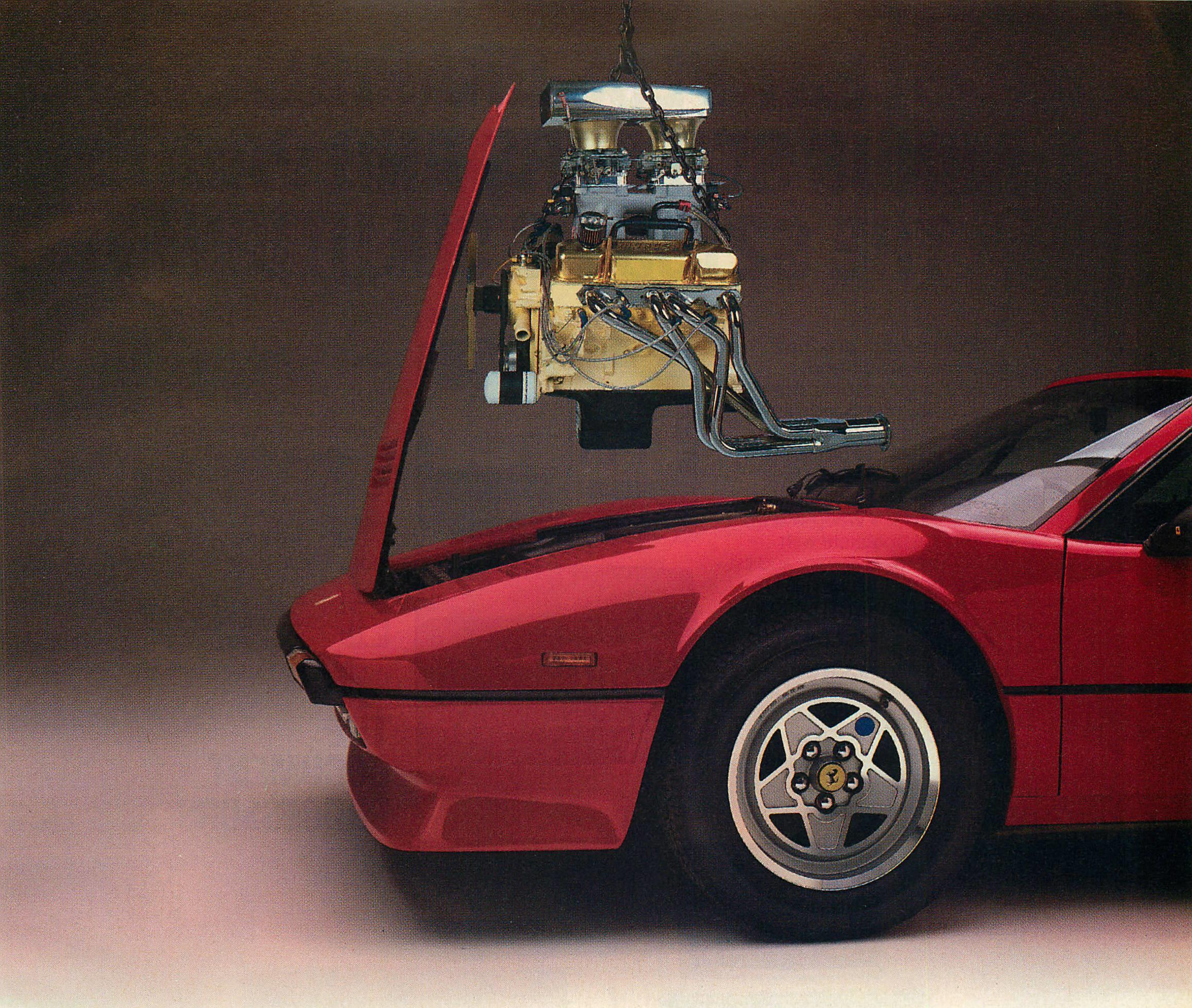
these options either may be set globally or attached to particular procedures. Although tracing can be performed only on commands in interpretive execution, the user can instruct ZIM temporarily to "forget" that certain procedures have been compiled by turning off the **runtime** option.

FULLY COMPILED APPLICATIONS

Procedures offer an important advantage over macros—they can be compiled into a faster-executing internal form using the ZIM compiler. Once compiled, they remain available for execution in compiled form, across sessions, until recompiled or explicitly marked *uncompiled.* Value-added resellers of ZIM applications will choose procedures over macros, because only by constructing a system of compiled procedures is a developer able to discard ZIM's interactive command processor and package his application with the smaller and more attractively priced ZIM runtime system.

Programs in either compiled or interpretive execution automatically use an 8087 or 80287 numeric coprocessor if one is present in the machine.

ZIM includes a comprehensive, integrated forms facility for interactive, formatted-screen data entry and editing. Forms are defined using ZIM's Interactive Form Definition Facility, invoked by the **form define** command. **Form define**'s layout mode allows the user to specify the appearance of a data entry form by "painting" its fixed and variable fields in the style of a full-screen editor. Its facility for defining *prompt fields* (fixed legends) can also be used to con-

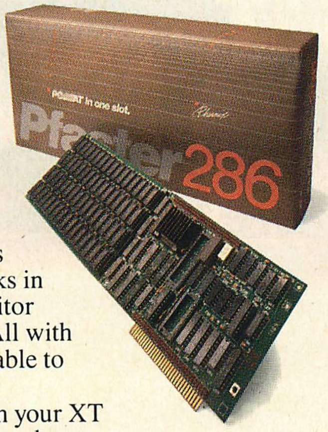


AT™ Pfantasies for your PC or XT.™

Want better speed and memory on your PC or XT without buying an AT?

You've got it!

Phoenix's new Pfaster™286 co-processor board turns your PC or XT into a high-speed engine 60 percent faster than an AT. Three times faster than an XT. It even supports PCs with third-party hard disks. But that's only the beginning.



You can handle spreadsheets and programs you never thought possible. Set up RAM disks in both 8088 and 80286 memory for linkage editor overlays or super-high-speed disk caching. All with Pfaster286's 1mb of standard RAM, expandable to 2mb, and dual-mode design.

You can develop 8086/186/286 software on your XT faster. Execute 95 percent of the application packages that run on the AT, excluding those that require fancy I/O capabilities your PC or XT hardware just isn't designed to handle. Queue multi-copy, multi-format print jobs for spooling. Or, switch to native 8088 mode to handle hardware-dependent programs and back again without rebooting. All with Pfaster286's compatible ROM software. And, Pfaster286 does the job unintrusively! No motherboard to exchange. No wires to solder. No chips to pull. Just plug it into a standard card slot, and type the magic word, "PFAST."

If you really didn't want an AT in the first place, just what it could do for you, call or write: Phoenix Computer Products Corp., 320 Norwood Park South, Norwood, MA 02062; (800) 344-7200. In Massachusetts, 617-762-5030.

Programmers' Pfantasies™
by
Phoenix

XT and AT are trademarks of International Business Machines Corporation. Pfaster286 and Programmers' Pfantasies are trademarks of Phoenix Computer Products Corporation.
For the Ferrari aficionado: yes, we know this is a rear engine car. We are showing the addition of a second engine to symbolize how Pfaster can be added to your PC or XT to increase performance.

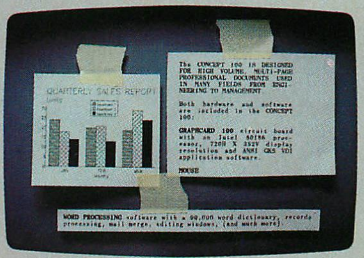
CIRCLE NO. 169 ON READER SERVICE CARD

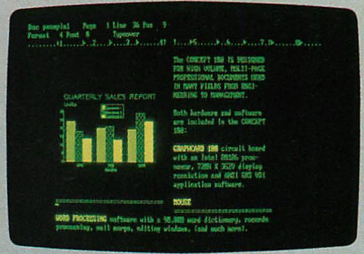
SEE US AT FALL COMDEX

TEXT PLUS GRAPHICS, MINUS SCISSORS AND PASTE: THE $2195 PC SOLUTION!

Now you can produce full-function text-and-graphics for reports, manuals, color overheads and slides right on your PC XT/AT. All you'll cut out is the scissors and paste! The CONCEPT 100 System lets you create text on a Wang-like word processor, complete with 90,000-word dictionary and list processing. You can easily create charts and drawings. Or import text or graphics from other systems. Then, use electronic cut-and-paste to combine them on-screen just as you'll see them in hard copy.

CONCEPT 100™ is based on the ANSI GKS* standard





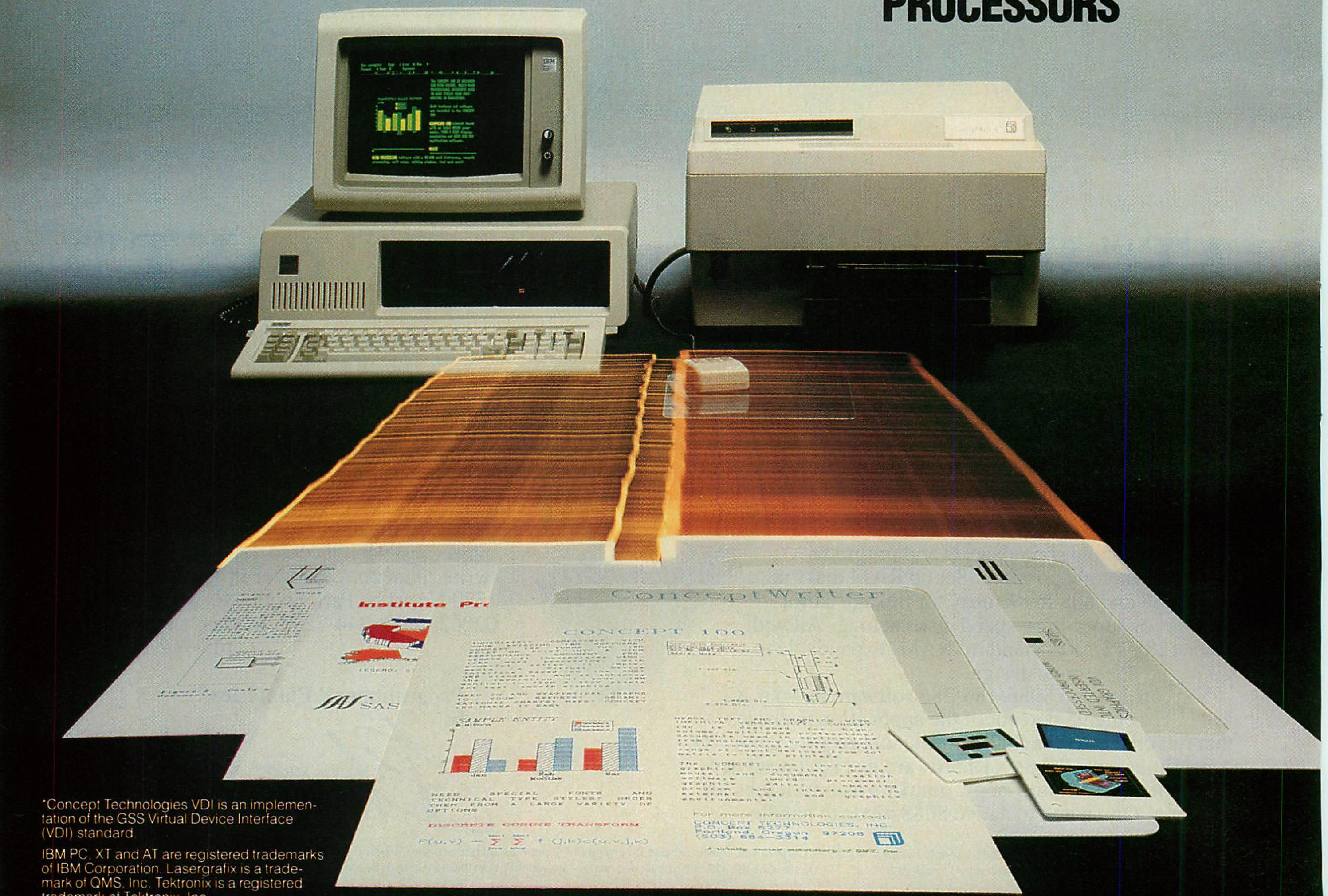
using the Virtual Device Interface. It offers Tektronix terminal emulation and output options like the ConceptWriter™ or Lasergrafix laser printers. In short, it's a personal publishing system at an unprecedented low price. Now available from your local distributor.

Call 1-800-631-2692 for more information.

CONCEPT TECHNOLOGIES, INC.
A QMS Company

P.O. Box 5277, Portland, OR 97208

THE DOCUMENT PROCESSORS©

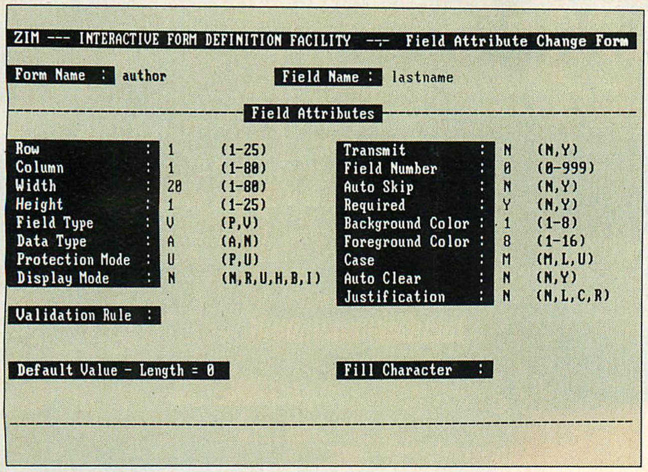


*Concept Technologies VDI is an implementation of the GSS Virtual Device Interface (VDI) standard.

IBM PC, XT and AT are registered trademarks of IBM Corporation. Lasergrafix is a trademark of QMS, Inc. Tektronix is a registered trademark of Tektronix, Inc.

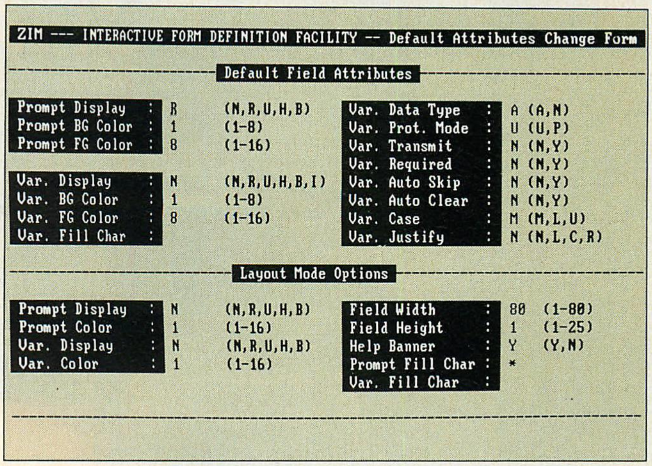
CIRCLE NO. 117 ON READER SERVICE CARD

PHOTO 1: *Field Definition Screen*



Each time the user paints a data entry field he is presented with a field definition screen. Using this screen he names the field and specifies its attributes from an extensive set.

PHOTO 2: *Default Attributes Change Form*



ZIM automatically assigns default values for field attributes, but these can be overridden during form definition if they do not conform to the needs of the application.

struct graphics using the PC's line-drawing graphics characters. Field heights as well as widths may be specified while painting variable data fields, allowing the construction of *box fields* in which *word wrap* is supported.

Data-entry fields are specified completely independent of the database definition; a form field need not be attached in any way to a field somewhere in the database. If a form field is given the same name as a database field, however, ZIM is able to use the name correspondence, reducing the specification burden on programmers writing ZIM programs to transfer data between forms and the database.

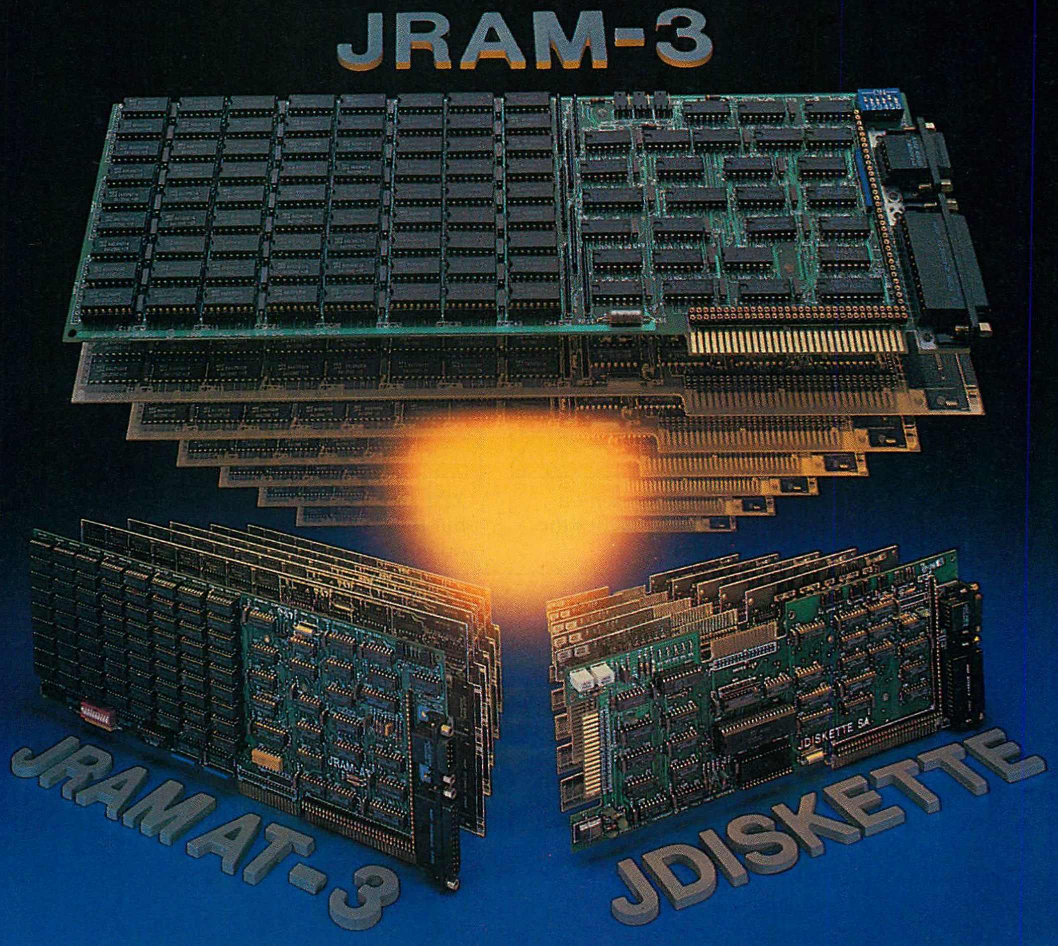
Each time the user paints a data-entry field, he is presented with a field definition screen (shown in photo 1). Using this screen, he names the field and specifies its attributes. The set of field attributes supported by ZIM is extensive. Fields may be protected, displayed in any of six display styles, and have default values and specific fill characters associated with them. Background and foreground colors may be specified for each field. Fields designated alphanumeric may have automatic upper- or lowercase conversion specified. Required or Optional attributes are supported. Fields may be given a Transmit attribute, which causes the forms manager to transfer control back to the user's program whenever data are keyed in the field.

Attributes for fields are given default values if the defaults are not explicitly overridden during form definition. The default values may be changed by the user if he finds that his conventions or those dictated by the requirements of a particular form differ from ZIM's preset defaults (photo 2).

At any time during interactive form definition, the user may strike a function key and toggle from layout mode into display mode. In display mode, the form is presented exactly as it will appear to the end user. While in this mode, the form designer can try out his field definitions by keying data into variable fields and watching how they are validated and displayed. Striking another function key toggles back into layout mode and allows the designer to move or adjust previously defined fields or to continue adding new fields.

A list of validation rules may be attached to a field. Validation rules for alphanumeric fields are specified as templates for legal input. A validation rule

THE FAMILY TREE



BRANCHING OUT

Tall Tree Systems presents JRAM-3, the newest member of the JRAM family. JRAM-3 is a fourth generation multifunction memory board and the successor of the highly praised JRAM-2. Designed to meet the latest expanded memory specification standard being implemented by the major spreadsheet vendors, JRAM-3 can access up to eight megabytes of memory for larger, more efficient spreadsheets. JRAM-3 can also be used for DOS memory, electronic disk, print spooler, and program swapping applications!

Determined to maintain our reputation as the price leader in memory expansion, Tall Tree Systems offers JRAM-3 fully populated with two megabytes for an amazing $699. A JETDRIVE/JSPOOL combo disk is included free of charge. This is the same highly acclaimed JRAM software that has helped make Tall Tree Systems the pioneer in the industry for bank switched memory and RAM disk technology. The new combo disk features JPAGER which allows any expanded memory application program to utilize multi-megabyte memory.

The family tree wouldn't be complete without an AT version. Look to us in September for the new JRAM AT-3, our third generation 16 bit board providing support for expanded spreadsheets. JRAM AT-3 will be available with two megabytes at $799.

Don't forget the rest of the JRAM family either. We'll continue producing our popular JRAM-2 and our present JRAM AT board, along with our full range of "slim daughter board" modules. The JRAM-2 and JRAM-3 can co-exist in the same computer, but only the JRAM-3 can be used for expanded memory applications. Use JRAM-2 for DOS, multi-tasking, print spooling, and electronic disk.

To complete the family line-up we feature JDISKETTE. This controller card lets you put 1.2 megabyte diskette drive(s) in your PC or XT. You can read, write, and format the AT high density diskettes and communicate easily between a PC or XT and an AT. You won't lose a single slot, since JDISKETTE replaces your old controller and handles up to four internal floppy drives. Free of charge with JDISKETTE, you will receive the popular JFORMAT software, a DOS compatible device driver that gives your computer the ability to handle expanded disk formats.

Welcome to the family!

CIRCLE NO. 197 ON READER SERVICE CARD

TALL TREE SYSTEMS
1120 San Antonio Road
Palo Alto, CA 94303
(415) 964-1980

enforcing the entry of parenthesized telephone area codes, for example, would read ([2-9] [0-1]9), in which the parentheses must be matched verbatim in the input, the numbers in square brackets indicate legal character ranges, and the last 9 is a wild card matching any number. A range of legal values may be specified for numeric fields.

Once a form has been created using **form define**, it is added to the data dictionary, just as entity sets and relationships are, and made available using the same **create** command that compiles definitions of other ZIM objects.

ZIM programs interact with the forms manager via an ingenious interface. A program paints the screen in preparation for operator data entry using the **form select** and **form display** commands. After that, the program passes control to the forms manager with the **form input** command, allowing the operator to edit data on the screen under control of the form specification. The forms manager retains control of the machine and continues to allow the operator to change information until he either types a Transmit or Exit key or keys data into a field that carries the Transmit attribute, at which point control is returned to ZIM.

Upon return from the **form input** command, information about the state in which the operator left the screen and the manner in which he sent it is available for inspection by the ZIM program in a set of predefined variables.

Within the ZIM program, all form fields in the currently selected form may be referenced by name and manipulated exactly as any variables are manipulated. The program can store values into form fields as well as reading values from them; fields into which the program stores values will be displayed with the revised values when the next **form display** command is given.

An even more useful characteristic of the forms manager/ZIM program interface is the ability of the ZIM program to regard the collection of data in a form's fields as though it were an entity set to which very fast access is possible. This ability makes it very simple to write ZIM programs that retrieve data from the database and display them on the screen or vice versa. A program to fetch an employee's information from the database and display it, formatted, on the screen could be as simple as:

```
find Employees where EmployeeNr = 1234
form select EmpForm
change EmpForm from current
form display
```

The form EmpForm plays a schizophrenic role in the previous example—in the **form select** command it identifies the screen definition developed earlier and saved using the **form define** facility. In the **change** command, however, EmpForm takes the place of an entity set and receives values as if it were part of the permanent database. In fact, a form's fields retain their values throughout a ZIM session, even when they are not selected or displayed, providing the ZIM programmer with the unusual ability to reselect and display a previously used form, complete with the field values it contained at the time of its last use.

Commands for manipulating forms allow a great deal of flexibility in managing the screen. The **noclear** option, which is available on the **form display** command, allows a program to stack multiple non-overlapping forms on the screen simultaneously. Another option, called **noprompt**, causes **form display** to show variable data only, allowing a program to paint a form, accept data from the operator, change a few fields, and refresh the data without ever flickering the screen or repainting its fixed legends. This option, used in conjunction with the Transmit field attribute, allows ZIM programs to implement the standard pop-up validation exchange with a data-entry operator, in which an entered field, such as a department number, is used by the program to retrieve the name of the department and immediately display it back to the operator, before he continues entering data elsewhere on the form.

Within the ZIM program, all form fields in the currently selected form may be referenced by name and manipulated exactly as variables are.

ZIM programs can dynamically override any of the field attributes in form definitions by using the **form set** command. **Form set** can also be used to change the set of keys with which the operator may transmit or exit a data-entry screen (Transmit keys apply all validation and required field rules to a screen before returning control to the ZIM program; Exit keys do not). Screen background and border colors can be changed dynamically using **form set**.

REPORT GENERATION

Consistent with ZIM's underlying philosophy, report specifications are nonprocedural. Creating a report specification with ZIM is more like painting a logical picture of a report than it is like writing a program to print one.

Reports are generated under ZIM by invoking a series of commands beginning with **report from** and ending with **endreport**. The **report** command's **from** phrase specifies a set from which report data are to be drawn. Commands appearing between **report from** and **endreport** must be reporting commands, although they may be entered just as any other interactive commands are entered. Report specifications are typically stored in document files.

The **report from** command may be accompanied by parameters indicating page and margin sizes and default column spacing. Parameters left unspecified are automatically assigned default values. Commands appearing in the body of a report specification allow the specification of report and page headers and footers, detail lines, and breaks (groupings of line items with a common value in some field or function of fields), which also may be bracketed by headers and footers.

Data for use in headers, footers, and line items can be specified in the form of database field names, constant character strings, variables, or expressions involving these. Each data item can be accompanied by an optional, COBOL-like format specification, or *layout* in ZIM parlance. Layout options for numeric and text data are extensive, allowing the standard treatments of negative money quantities (leading or trailing minus, parentheses, *CR* suffix), floating dollar signs, embedded commas, leading asterisk fill, and so on. Subtotals, totals, and counts can be accumulated and presented in a report as headers, footers, or line items.

Complex report specifications tend to be verbose because layout specifications must be given individually for each data item appearing in a report. A line item including ten fields, for example, would require ten data reference/layout terms instead of a list of ten data items followed by a single layout specification giving the format of all ten. Frequently, absolute column positions must be specified in order to align adjoining field layouts. New-line and new-page operations may be specified to oc-

CIRCLE NO. 136 ON READER SERVICE CARD

cur before line or heading/footing items but not after, restricting the report designer's formatting capabilities. Nonetheless, the report facility is comprehensive and allows the composition of professional-looking reports.

A ZIM report specification that creates a report listing the articles per author, complete with each author's compensation totaled for each year, is shown in listing 2.

Used at the direct, interactive command level, ZIM provides support for answering ad hoc queries against a database. ZIM lends itself quite well to interactive, ad hoc use because it embodies the philosophy that users should specify, wherever possible, what they want nonprocedurally; they always should be able to focus on the *what* rather than on the *how*.

Typical queries requiring the selection of a small subset of data from a much larger collection can be accomplished quickly by using the **list** command with a where clause that gives the selection condition. **List**'s format modifier allows the user to pare down output by removing any irrelevant fields, as in the following example:

```
list Employees where Age >= 60 format
   LastName FirstName Salary.
```

Even with the command macro facility, nonprogramming users may have trouble composing queries and saving their frequently used query sequences in document files for reuse. It does require that they learn the use of a text editor, the interactive command language, and ZIM's simple mechanism for passing parameters to macros.

ZIM's **find** command for constructing aggregations of data in sets can be used to great advantage during interactive sessions because it allows the user to factor complex queries into a set of simple, more easily composed steps. Where the query

```
list Authors Write Articles PartOf Issues
Where Author.State = NY and
   Issues.Year = 1985
Sorted by (Payment + Bonus) Descending
format LastName FirstName AuthorNr Title
   Payment Bonus
```

might intimidate a nontechnical user, the equivalent sequence of commands:

```
find Authors where State = NY -> Authset
find Authset Write Articles -> AuthArt
find AuthArt PartOf Issues Where
   Year = 1985
sort by (Payment + Bonus) Descending
list format LastName FirstName AuthorNr
   Title Payment Bonus
```

goes at the task stepwise and is more easily digested at first glance.

In this example, the power of the entity-relationship model is apparent. Data from several entity sets were culled together with a minimum of specification work by giving a series of English-like commands. Technically inclined ZIM users may grasp the concepts of the entity-relationship model and structure data and frame queries such as the above example. Nontechnical users, however, may find they have to study more than they are accustomed to before becoming comfortable with ZIM at the direct command level.

Find can be used to commission the construction of very large aggregations of data spanning many entity sets and relationships. Depending on the size of the database and the composition of the "found" set, the set-building process can be very time-consuming. A user who inadvertently starts a **find** to build a very large set, then realizes his mistake and attempts to interrupt the command, may encounter an annoying delay. ZIM is not attentive to the terminal while performing operations that do not cause terminal input and output, and it sometimes imposes unpredictably long delays before recognizing that the user has typed an interrupt character.

ZIM provides no on-line help at the interactive command level, so a first-time end user has no choice but to begin with a careful reading of at least part of the manual and, possibly, a look through the on-line tutorial. The command processor's diagnostic error messages, however, are comprehensive, and usually point quite specifically to the problem with an errant command.

PACKAGED ZIM APPLICATIONS

Developers can package command-oriented, forms-oriented, or hybrid applications under ZIM. A command-oriented application is built by developing a library of ZIM procedures or macros that the user invokes from the interactive command level. Hybrid applications incorporate both command and forms-based interaction with the user. Fully packaged applications converse with the user only by way of formatted screens, thus hiding the interactive command level entirely.

ZIM's applications development facilities are sufficiently flexible that a developer can achieve virtually the same end product in ZIM that he could achieve by programming in a general-purpose language. ZIM is not intrusive; an end user of a well-packaged, forms-based application never "sees" ZIM except when he is using DOS commands to back up his database.

This development flexibility stems almost entirely from the generality of the forms management system and the ability of ZIM programs to take control away from the operator momentarily as his cursor leaves any data field shown on the screen. This transmit-field capability permits the applications builder to adjust the database or the screen by employing arbitrary program logic at any point during an operator's use of the screen. Any validation or interaction that cannot be accomplished automatically by the forms manager can be supplied by the controlling applications program. Even though ZIM itself provides no on-line help, for example, a ZIM-based application can be built to include whatever level of on-line help the developer might desire.

End users will not be surprised or confused by eccentric management of the PC's screen or keyboard by the forms manager. Standard meanings and actions are attached to the keys on the cursor control keypad, the Tab and Enter keys, and so on.

Installing ZIM on a hard-disk system entails copying two diskettes into a directory, copying a special file containing error messages into the root directory, adjusting the CONFIG.SYS file to specify ANSI terminal management (DEVICE=ANSI.SYS) and allowing for a minimum of 20 open files (FILES=20). The installation process can be completed in less than five minutes.

An empty database is established by creating a directory and running the **ziminit** program; this is the point at which the user must decide upon a basic disk page size. The guidelines for choosing a page size are simple to follow: applications that spend the bulk of their time performing random disk access to accomplish keyed look-ups perform best with a small page size (the minimum size is 1,024 bytes); applications that perform sequential processing benefit from a larger page setting. The maximum disk page size is 8,192 bytes. No installation procedure is necessary except for the changes to the CONFIG.SYS file and the initialization of database directories for floppy disk systems. The process of database directory initialization takes a minute or two on hard-disk systems.

Users may, optionally, create a ZIM-specific configuration file named CONFIG.ZIM. This file, structured like DOS's CONFIG.SYS, may contain parameter specifications with which the user can tune his system to achieve better per-

TABLE 1: *Benchmark Results*

BENCHMARK TASK	TIME (secs)
Add 900 records to an empty database	246
Index database on two fields (7 bytes)	—
Document and tally codes from one field	25
Mass change of one field (28 records of 900)	4
Extract selected records to create a text file	6

ZIM performs basic operations, such as retrieving, importing, and sorting data, at a lively pace that is competitive with other data managers. More subjectively, it feels adequately fast, though not dazzlingly so, in performing operations visible to the end user, such as displaying a formatted screen.

formance or to meet unusual requirements. The number of buffers ZIM uses during sorting, for example, can be set. The number of files ZIM keeps open even when not in active use in its open file cache can be adjusted. Certain limits, such as the maximum number of parameters a procedure can receive (default limit 20, maximum limit 256), can be relaxed by making entries in CONFIG.ZIM. Error-trace mode may be set, causing ZIM to log all error messages to a disk file in addition to displaying them at the terminal.

ZIM databases are backed up to secondary storage using standard DOS facilities. Using the DOS COPY or BACKUP commands to save a copy of all files matching the name template ZIM*.* ensures that all files in a database (excluding document files that were named by the user) are captured on the back-up media.

A ZIM database can be corrupted if the user switches the machine off without properly leaving the program. Improper termination may or may not cause parts of the database to be left in an inconsistent and unusable state. If a database is damaged, recovery requires reloading the database from back-up storage; reloads must restore an entire, consistent database copy.

ZIM loads and runs quickly. Its copy-protection scheme is somewhat bothersome for users with hard disks—it requires that one of its release disks be present and write enabled in diskette drive A: whenever ZIM is invoked. A specially named profile document containing ZIM commands is executed each time ZIM is invoked; the user can edit this document to configure a particular user's defaults or to start an application running automatically. ZIM obeys standard rules for interfacing with DOS, and it can coexist with programs such as Borland International's SideKick, provided that the system contains enough memory to run them.

ZIM's reference manual is complete, clear, and well-organized. The manual is accompanied by a *User Guide and Tutorial*, which consists primarily of hard-copy versions of the screens presented by its interactive tutorial, but also includes a pithy introduction to ZIM's primitive data elements and to the concepts underlying the entity-relationship data model and the unusual set vehicle for aggregating data. System developers will find the document lucid and indispensable; users inexperienced in computing will find it tough going.

The ZIM package also includes two other forms of documentation, in a less conventional sense of the term. A disk containing a sample application is provided. Running the sample system and thoroughly inspecting its programs and form definitions will provide a wealth of information to the new user. Another disk included in the package contains the *ZIM Interactive Tutorial*, which describes fundamentals and commands step by step and allows the viewer to try some simple commands. The interactive tutorial is a good starting point for a new user, but it cannot stand alone as an introduction to ZIM.

COMPETITIVE PERFORMANCE

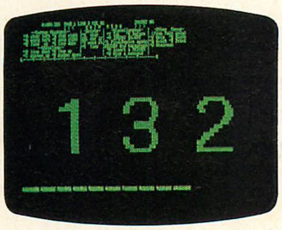
ZIM performs basic operations, such as retrieving, importing, and sorting data, at a lively pace that is shown to be competitive by the *PC Tech Journal* benchmark timings (table 1). More subjectively, it feels adequately fast, although not dazzlingly so, in performing operations visible to the end user, such as displaying a formatted screen.

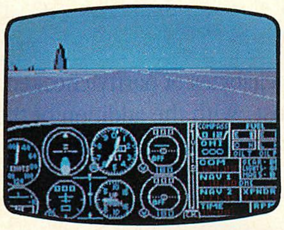
ZIM incorporates a strategy analyzer to accelerate the answering of queries involving indexed fields. In queries, indexes are used in applying both equality and inequality tests, such as *greater than* or *less than or equal*. Indexes are used in processing compound conditions as well, such as Age > 30 AND LastName = 'Smith'.

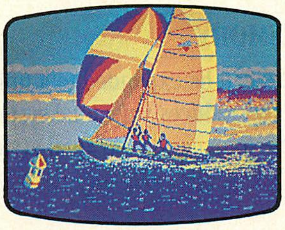
DOES YOUR HI-RES GRAPHICS BOARD SHOW YOU ONLY PART OF A SPREADSHEE

When it comes to printing out spreadsheets, most IBM®-compatible graphics boards come up short.

But not the Genoa Spectrum™ Color/Mono/Graphics Board.







The Spectrum Color/Graphics Board is the most compatible board you can buy. Which means it runs the most software. Including Flight Simulator,™ Lotus® 1-2-3,™ Symphony,™ SideKick,™ PC Paint,™ and much, much more.

It goes all the way to 132 columns. The largest text display of any graphics board you can buy. There's even a special driver that'll show you 132 columns of Lotus, which normally only has 80.

And together with your 132-column printer, the Spectrum can also print every one of those 132 columns. So you can see more of what you're doing. On the screen. And on paper.

But the best part is, you don't have to pay more to do it. Here's why:

Up until now, if you wanted both graphics and color, you had to buy two boards. One for hi-res mono graphics, one for color.

But the Genoa Spectrum puts color and graphics *on the same board*. So you save yourself a slot. And about $250.

You can also save yourself the price of a color monitor. Because Spectrum will emulate your color software in monochrome. And for your monochrome software there's a 720 x 348 hi-res mode.

So either way, you're covered.

Spectrum is compatible with all color and monochrome monitors. So you'll always get 16 beautiful colors in color. Or 16 shades of monochrome in mono.

For all the technical specs or the Genoa Spectrum dealer nearest you, just call us at 408-945-9720. Or write Genoa Systems Corporation, 73 E. Trimble Road, San Jose, CA 95131.

The Genoa Spectrum. It's the only board you'll ever need for color, graphics, and full size output.

Anything else is either too much money or incomple

Genoa

We make IBM PCs better.

CIRCLE NO. 209 ON READER SERVICE CARD

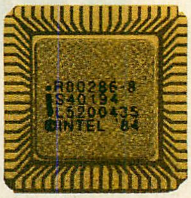
Fast! SYSTEM 286

33% Faster than IBM's Powerful PC AT

SYSTEM 286 STARTING AT $2,495.00

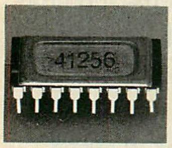
8 MHz CPU,
512K RAM,
Parallel Port,
Winchester Controller,
Floppy Controller,
1.2 MB Floppy,
83 Key Keyboard,
Battery Backup
Clock Calendar

THE ALR SYSTEM 286



The ALR System 286 is based on the advanced 80286-8 16 bit microprocessor with a system clock rate of 8MHz. This performance results in running most IBM® PC or XT software up to 350% faster: In seconds you'll be able to recalculate large spreadsheets and instantly load files.

Compatibility by Design. At Advanced Logic Research, the System 286 was designed to be 100% IBM PC AT bus compatible with full attention paid to the BIOS ROMs resulting in total support of IBM DOS 3.0, 3.1 and Xenix™ operating systems.



2MEGA BYTE USER MEMORY

The System 286 CPU Board can be expanded to a full 2 megabytes of parity checked user memory. This saves you valuable expansion slots and reduces your total system cost. With the addition of the award winning ALR Challenger you will be able to expand your total system memory to an incredible 15 megabytes.

ADVANCED LOGIC RESEARCH, INC.

ALR is an engineering and manufacturing company to OEMs and system integrators with the following enhanced CPUs and controller products for the IBM PC:

CHALLENGER! ALR System 286, IBM PC AT Compatible Multifunction Card with 4 Megabytes of RAM, 2 Serial Ports, Printer Port and Game Port.

I/O FLOPPY: Floppy Controller with Clock Calendar, Serial Port and Printer Port for the Advanced PC, XT or IBM PC.

FAST 286: 8 Mhz, 2 Megabyte IBM AT Compatible

ADVANCED PC TURBO: 8 Mhz, 1 Megabyte IBM PC Compatible

ADVANCED XT TURBO: 8 Mhz, 1 Megabyte IBM XT Compatible

QUATTRO PLUS: Short Card, 4/8 Channel Serial Ports and Printer Port for the ALR/IBM PC, XT or AT.

For more information on a dealer near you, contact ALR and let's talk about your FAST! System 286.

COMDEX/Fall '85
Booth 3474 (Weltec)

XENIX™ is a registered trademark of MicroSoft Corporation
IBM® is a registered trademark of IBM Corporation

ALR

ADVANCED LOGIC RESEARCH, INC.
2991 E. WHITE STAR AVE.
ANAHEIM, CA 92806
(714) 666-2951

CIRCLE NO. 116 ON READER SERVICE CARD

ZIM's query optimization does not extend into its set-building facilities. Until they gain a key insight into the internal workings of ZIM, inexperienced users will be surprised at the length of time some seemingly simple queries take to run. For example, ZIM satisfies the following query

```
list all Employees WorkIn Departments
   where Employees.Age > 65
   sorted by Employees.Salary
```

by first building the set, Employees WorkIn Departments; then applying the selection condition to the set; and, finally, sorting the resulting subset. If the Employees entity set is large, the construction of Employees WorkIn Departments can be very time-consuming. In this case, the user would get a much quicker answer by doing some query optimization of his own. If the number of employees over the age of 65 is small relative to the total number of employees, the user would receive a much quicker answer by posing the query this way:

```
find all Employees where Age > 65 ->
   empset
list empset WorkIn Departments sorted by
   Employees.Salary
```

The order in which ZIM evaluates queries is documented clearly in the user manual. The performance implications of the evaluation order are not clearly set out, however, and users must undergo a process of experimentation and discovery before they are able to take full advantage of ZIM's speed.

ZIM is a well-conceived, soundly implemented, thoroughly professional system. Its design evidences a strong commitment to consistency and to the goal of natural, nonprocedural user interaction on Zanthe Information's part. Its implementation evidences its developers' considerable skill and experience. Despite its relatively recent appearance on the data managers' market, ZIM has the feel of a mature product because of its attention to the practical requirements that arise when managing large databases. It imposes no unreasonable restrictions on users or applications developers, is configurable in most useful respects, and embraces the harsh realities that disks break, users err, and programs go astray.

End users will enjoy ZIM's complete array of functions and its competitive performance. Although it can be effective at the interactive command level in the hands of any PC user with data management requirements, ZIM will present a significant challenge to new users who lack a strong grounding in data management technologies. Learning to use ZIM effectively requires an investment of time in study even for technically sophisticated users.

ZIM is likely to find its most widespread application in settings in which computing professionals build packaged turnkey systems or partially packaged

ZIM's design evidences a strong commitment to consistency and to the goal of natural, nonprocedural user interaction on the part of Zanthe Information.

systems for redistribution or resale—for example, in corporations where a central services group distributes information-processing facilities to operating units or in vertical software markets served by systems houses or independent developers. Zanthe's price structure creates a powerful incentive for redistributors to build complete packages that do not require ZIM's interactive command processor.

From an applications developer's viewpoint, ZIM's fourth generation system-building approach offers an ingenious, integrated forms management facility, a modern, competent general-purpose programming language, and an adequate report generator.

Any organization in which ZIM gains a foothold is likely to benefit from ZIM's advanced data model. The experience of casting database definitions under the semantically rich entity-relationship scheme undoubtedly will convert many designers. The model provides an excellent framework in which to establish and document the structure of a database, even if the ultimate system implementation is to be performed using data management facilities based on another data model.

ZIM's complexity and youthfulness have left it with a few rough spots; nonetheless, it provides a powerful vehicle for developing PC data management applications and merits attention from any system developer with data management requirements.

Richard M. Foard is vice president of software development for ROADNET Systems Corp.

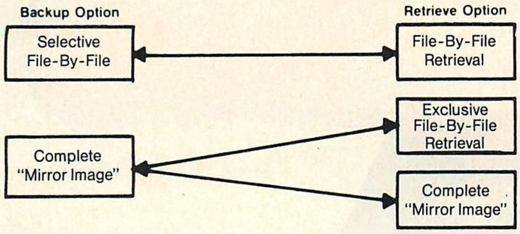
LISTING 1: PCNTPGS

```
% File:  pcntpgs
% Auth:  Richard Foard
%
% Zim procedure to discover the number of pages booked for an
% issue; gives total numbers of editorial and listing pages.
%

Procedure pcntpgs(in vol, in num)

  % validate input, fetch indicated issue
  let vvolume = $tonumber(vol, 0)
  let vnumber = $tonumber(num, 0)
  find Issues where Volume = vvolume and Number = vnumber

  % if valid issue, get articles, list titles, and count pages
  If $SetCount > 0
        let vtotedit = 0          % total editorial pages
        let vtotlist = 0          % total listing pages

        find all Articles where Volume = vvolume \
                       and Number = vnumber
        while $SetCount > 0
          let vtitle = Title
          output vtitle

          let vtotedit = vtotedit + EditPages
          let vtotlist = vtotlist + ListPages

          next
          let $SetCount = $SetCount - 1
    endwhile
        output
        output 'Total pages:'
        output   vtotedit ' Editorial Pages'
        output   vtotlist ' Listing Pages'
        output   (vtotedit + vtotlist) ' Total Pages'
  else
          output 'Issue' vvolume '/' vnumber 'does not exist'
  endif
Endprocedure
```

LISTING 2: AUTHORS.RPT

```
report from Authors Write Articles PartOf Issues \
  where State=AZ or State=SD \
  sorted by LastName FirstName Volume Number

report heading \
  'Articles by Author'          : line 7 center:

page heading break heading level 1 \
  'Pmt  Vol Num                Title'

break 1 write.authornr \
  heading \
  $concat($chop(firstname), ' ', $chop(lastname)) \
  : newline 2 \
  mask 'Articles by: ??????????????????????????????':

break 2 year \
  heading \
  ''               : newline 2: \
  footing \
  year    \
  : newline 2 mask '     Total compensation in year ????': \
  $money($total(payment+bonus)) \
  : column 38 mask '$$$,$$$.99':

detail line \
  column heading off \
  (payment + bonus)                 : mask 'ZZZZ':\
  issues.volume                     : column 6 mask 'ZZ':\
  issues.number                     : column 9 mask 'ZZ':\
  title                                 : column 12:

endreport
```

Innovative Backup and Hard Disk Drive Systems For Your IBM PC, XT and AT



The Standard of Excellence in Backup Software

- Revolutionary Everex Backup systems give you the backup/retrieve flexibility you want while saving you hours of time. You can backup your hard disk drive in minutes with a fast "mirror image," then retrieve this information either file-by-file or the entire hard disk image.
- High speed backup/retrieve—up to 5MB per minute.
- Menu Driven software includes the largest selection of file selectable options—choose from name, date, time interval, global, qualifier and more.
- Advanced installation program automatically configures the optimal backup/retrieve speed with your hard disk drive.

The Leader in External Expansion Systems

- All external systems are available with any combination of Everex Backup and Hard Disk Drive systems.
- Slimline systems include one short and three long expansion slots for adding more accessory boards.
- Half-Size system includes four long expansion slots.
- Full-Size system (looks like your PC) includes eight long expansion slots.
- External systems include one high quality, round shielded cable.

The Total Solution For Backup That Plugs Right Into Your Computer

- A wide selection of backup solutions with the price and performance to meet your needs:
- The EXCEL 4500, 45 and 60 MB high performance ¼-inch Streaming Tape systems, the EXCEL 200, 20 MB Cassette system and the EXCEL 10 MB Floppy Tape system.
- Space-saving half height units with single board controller.
- Combine with Everex hard disk drives for lower cost and higher performance.
- Unique "piggyback" power supplies ensure dependable operation and save space.

Visit your local Everex dealer today and ask to see Everex products in action. For the name of your nearest Everex dealer, please call (415) 498-1111.

Imagineering Ultimo, Australia TLX: 74349 IMAGIN AA
Microage Distribution Ltd. London, England TLX: 881 3241 WONGS G
Feeder Paris, France TLX: 4413241 FEEDER
Automated Office Systems Hout Bay, South Africa 2721-70-8091
Survex, 1027 Speers Road, Oakville, Ontario Canada L6L-2X5, 416-842-6093
Pride Computers, 102-8167 Main Street, Vancouver, British Columbia, V5X 3L2, 604-321-5690

EVEREX
EVER for EXcellence

Address: 47777 Warm Springs Blvd., Fremont, CA 94539 (415) 498-1111.

Dealer Hotline (800) 821-0806 • In CA (800) 821-0807

Back up 10 megabytes in 8 minutes! (or less)

with Fastback

If you've ever lost valuable data, you know backing up your hard disk is a *MUST.* For your business, this necessity can become an expensive nightmare. Until now. Fastback™ (Version 5.0) is the hard disk backup (and restore) software utility that eliminates the need for expensive and time consuming tape backup systems. It's so fast you can back up a full 10-megabyte hard disk on standard 5¼″ floppies in less than 8 minutes or 10 megabytes of data on the IBM™ PC-AT in less than 4 minutes! This can mean a 95% savings of time and money.

Fastback also gives you an added edge that no other backup system can boast. It uses advanced *ERROR CORRECTION* techniques to recover data from damaged diskettes. So if your disk is damaged after the backup procedure, Fastback (Version 5.0) will still be able to recover the data.

Fastback is fully self-contained and functions with any make or size hard disk. It works with PC-DOS™, or MS-DOS™ version 2.0 or higher and requires no additional hardware to take up valuable expansion slots in your IBM PC/XT/AT or compatible.

Fastback can automatically format your floppy disk as it saves your data; both drives will be used on a dual drive system to eliminate the disk change time. Fastback cataloging feature provides rapid lookup of archived data and prevents accidental overwriting.

You control what files you want because Fastback is completely file oriented, not an image mode backup.

All this for just: **$179.**

With Fastback, backing up is fast and easy. Remember, among those who backup regularly, there are only two types—those who use Fastback and those who will!

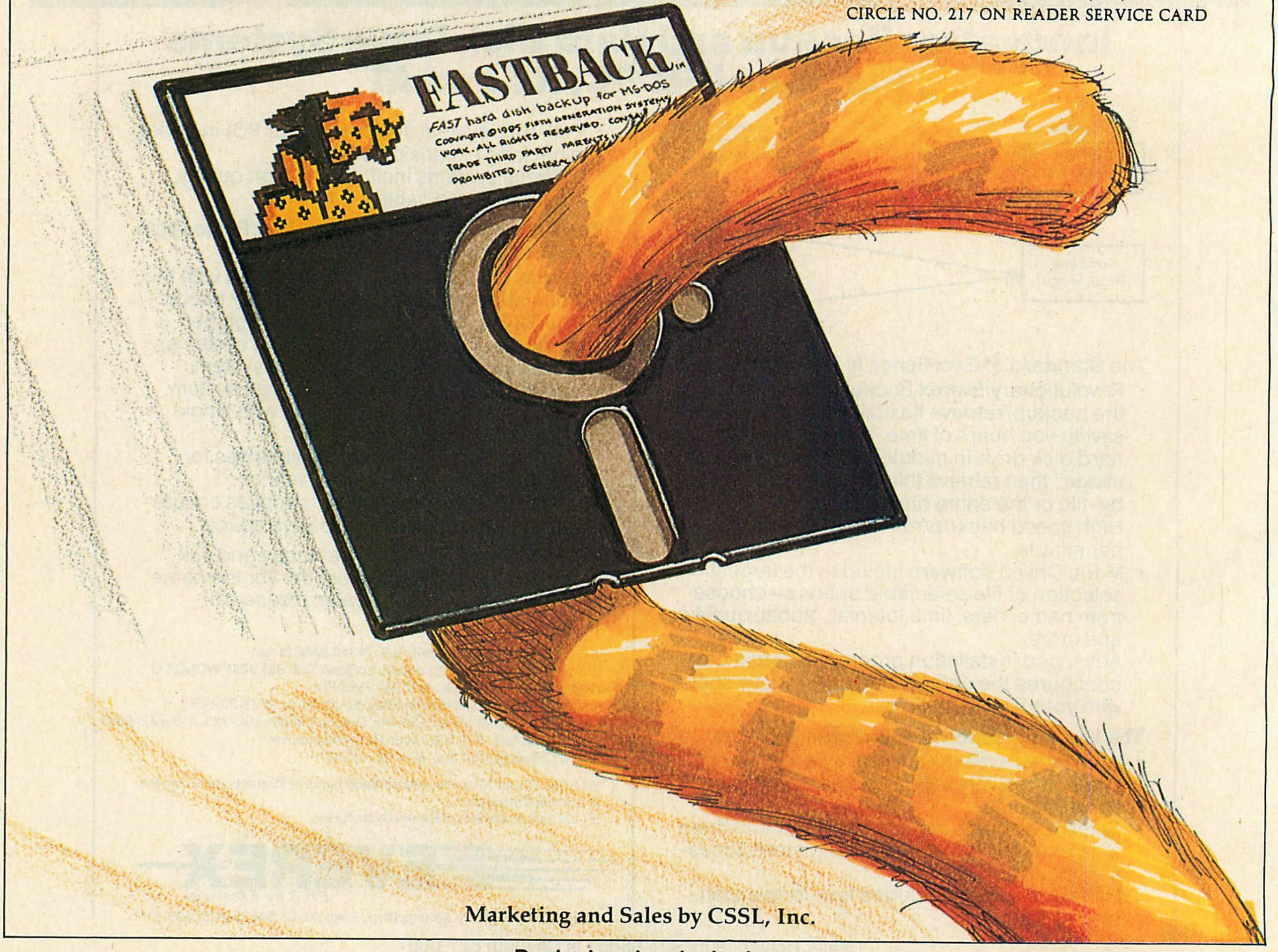
For further information, call or write to:

Fifth Generation Systems

7942 Picardy Avenue • Baton Rouge • LA • 70809

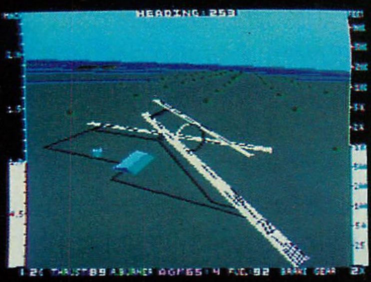
Telephone: (213) 439-2191

CIRCLE NO. 217 ON READER SERVICE CARD

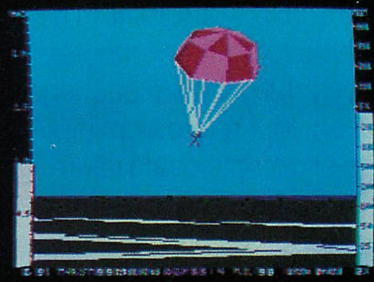


Marketing and Sales by CSSL, Inc.

Dealer Inquires Invited



View from F-16 Cockpit



Ejection from F-16 (Control Tower View)

Enjoy Microsoft Flight Simulator Scenery

Jet

From the author of Microsoft Flight Simulator comes a new dimension in flight simulation. Jet simulates two fast and maneuverable supersonic jet fighters, a land-based F-16 or a carrier-based F-18. The Jet simulation includes modern electronic flight instrumentation and the most advanced weaponry available. Jet's combination of simulation sophistication, excellent visual attitude references, and fast animation speed makes it truly enjoyable to fly.

Excellent aircraft control plus ballistic thrust gives you the kind of aerobatic maneuverability only a modern jet fighter can provide. The attitude indicator is easy to read no matter what your orientation. This indicator, combined with a full-screen out-the-window view, helps you get the most out of Jet's excellent flight controls. An important consideration when traveling at speeds in excess of 1300 MPH.

With Jet you can fly through either structured on non-structured environments. Engage in a deadly variety of combat missions. Explore the wargame territory, or relax by practicing precision aerobatic maneuvers. Load in scenery from optional United States Scenery Disks. You can even load in scenery off the Microsoft Flight Simulator disk.

Jet's technology incorporates our newest high-speed graphic drivers, and introduces new scenery concepts for improved viewing on non-color and RGB monitors.

Jet requires an IBM™ PC, PC jr, PC AT, Compaq™, or Tandy™ 1000-series computer, with 128K minimum memory and the Color/Graphics Monitor Adapter, Hercules™ Graphics Card, IBM Enhanced Graphics Adapter, or Sublogic High-Speed Graphics Card.

See Your Dealer . . .
or write or call for more information. For direct orders please enclose $49.95 plus $2.00 for shipping and specify UPS or first class mail delivery. Visa, MasterCard, American Express, and Diners Club cards accepted.

SCENERY DISKS FOR MICROSOFT FLIGHT SIMULATOR AND JET ARE NOW AVAILABLE.

IBM is a trademark of International Business Machines Corp.
Compaq is a trademark of Compaq Computer Corp.
Tandy is a trademark of Tandy Corp.
Hercules is a trademark of Hercules Computer Technology.

subLOGIC Corporation
713 Edgebrook Drive
Champaign IL 61820
(217) 359-8482 Telex: 206995

Order Line: (800) 637-4983
(except in Illinois, Alaska, and Hawaii)

CIRCLE NO. 145 ON READER SERVICE CARD

Keyboard Shortcuts

Of the six keyboard macro processing programs reviewed here, five are recommended.

JOHN WALKENBACH

You press the button, we do the rest. This expression, which Kodak used a few decades ago to describe the ease of operating its cameras, also explains much of the appeal of personal computers. Most computer users know the feeling of satisfaction that accompanies the final keystroke to recalculate a complex spreadsheet, compile a program for the final time, or locate some critical records in a database. Unfortunately, typing the sequence of keystrokes leading up to that one final, satisfying keystroke often can be pure drudgery. Keyboard macros allow the user to have many of those preliminary keystrokes done automatically, thereby eliminating the drudgery and errors that normally occur when the sequence is typed manually.

A macro is a shorthand way of expressing a more complex function. For

ILLUSTRATION • FRANK RILEY

example, in this article, the phrase *macro processor* appears frequently. A macro, represented by a key combination such as Alt m, could be used to avoid having to type that phrase each time it is needed; every time the combination Alt m is typed, the words *macro processor* will appear on the monitor. Alt m is a macro, and the phrase *macro processor* is the expansion of the macro invoked by the Alt m keystroke.

This is a simple example; macros can be made much more complex. A macro can be established to perform multiple functions and even to generate complex help screens or menus—all invoked by a single keystroke.

Many software products—SuperCalc, 1-2-3, WordPerfect, Smartcom II, and IBM's Personal Editor, to name a few—include built-in macro capabilities. IBM's BASICA language has 22 nonprogrammable macros that expand into BASIC keywords and that can be invoked by pressing the Alt key with one of the letters. In addition, using BASIC's KEY statement, the ten function keys can be programmed with a macro expansion of up to 15 characters.

Separate macro processors are also available and are designed to work in conjunction with other software. A number of such products are reviewed here: KEYSWAP, Keyworks, ProKey, RE/Call, SmartKey, and SuperKey.

Keyboard macro processors have four general uses. One of the most common is to simplify repetitive input. In word processing or programming applications, macros save a great deal of time by preventing the user from typing long or complex character strings repeatedly. For example, a user could define a macro that expands into the name of his company. Pressing Alt c is much simpler than typing "Consolidated Computer Hardware Corporation of America." Macros can also be defined to serve as boilerplates for contracts or frequently used letters.

A user-input feature available with most macro processor programs allows parameters to be passed to the macros. This makes program inputting and formatting a breeze. A macro, say Alt i, could be defined as the following:

```
If xxxx <CR>
  then xxxx <CR>
  else xxxx <CR>
```

Invoking the macro with Alt i causes "If" to appear on the screen, followed by a pause. The "xxxx" represents user input—the fill-in-the-blank portion of the macro. After the user enters his input, the macro resumes by typing "then" and stops for more user input. Finally, the macro displays "else" and waits for the final user input. Indenting, spacing, and carriage returns are all a part of the macro definition and are carried out automatically.

The second general use for keyboard macro processors is to simplify or customize various program commands. A particular word processing task that requires changing defaults, for example, is a perfect candidate for a macro processor. The following series of WordStar commands sets the margins to 5 and 76, turns off right justification, and invokes double spacing:

```
^OL5<CR>^OR76<CR>^OJ^OS2
```

This entire sequence of commands could be put into a macro and invoked with a single keystroke.

Macro processors also allow users to customize commands so they are easier to remember or more consistent with other software packages. In WordStar, the backspace key can be redefined so that it deletes the character to the left of the cursor, which is the normal action for this key in most software. Macros also can be used as an approach to standardization of software. For example, with macros the F1 key could be set up to bring up the help screen in all applications.

> ***Most macro processors allow that definition files be read or saved from within an application.***

In addition, many macro processors allow users to rearrange the keyboard. For example, the slash key on the lower left of the keyboard can be made into a shift key. It is also possible to relocate the cursor control keys in order to allow simultaneous use of the numeric keypad without having to switch modes with the NumLock key. Many spreadsheet users take advantage of a macro program to move the cursor-control keys to the function keys, which allows them to use the numeric keypad for input and the function keys to move from cell to cell. Furthermore, for users who prefer the Dvorak keyboard, some macro processor programs include this layout.

Finally, macros can be used to set up turnkey systems for beginners or those not willing to learn all the intricate commands of an application package. With a single keystroke, a well-constructed macro can start up a program, read additional files, change the defaults, and prompt the user for input. Some macro processor programs even allow the user to develop menus and help windows that can be called to the screen from within any program. The macro processor saves the current screen, pops up a window, and restores the original screen when the window is no longer needed.

THE MAGIC BEHIND MACROS

All macro processors are resident programs that attach themselves to DOS (using interrupt 27H); they remain in memory at all times but are dormant until needed by the user. In addition to the memory that is required for the program's resident code, macro processors require a buffer area in memory to use for storing the macro names and expansions. Therefore, using a macro processor will decrease the amount of available memory in a system by the length of the program plus the buffer space. If more memory is needed, however, most of these products can be uninstalled and the memory can be released for other applications.

IBM PC programs get keyboard input by using the DOS service calls or the ROM BIOS functions. Most commercial programs intercept the BIOS keyboard interrupt 9H and avoid the additional overhead of going through DOS.

Macro processors intercept the keyboard input and determine if the key just pressed has been assigned a macro expansion. If a macro has been assigned, the macro processor substitutes the appropriate string of characters and sends them to the application.

Macro definitions are stored in files that are read into memory in preparation for a specific application. Thus the user may maintain separate sets of macros for each application. When the user exits the application, he can load a new set of macros (for example, macros for DOS commands). Usually, this is accomplished automatically with batch files. Most macro processors allow definition files to be read or saved from within an application. This is handy if a user is running one application, such as WordStar, and discovers that the macro set in memory is for another application, such as 1-2-3. Having separate sets of macros for each application allows each set to use the maximum buffer space and permits the standardization of keys from application to application.

A macro may be defined in different ways. In the *record mode* method, one command is typed to begin recording keystrokes and another to stop recording them. Anything typed between these two commands is included in the macro, including all Backspace, Enter, and cursor-control commands. The advantage here is that the macro can be tested as it is being defined.

Another method of defining a macro is with an online, full-screen editor, which is becoming common in the newer macro processor programs; all of the macro processors reviewed here, except ANSI.SYS, include an editor. These editors can be brought up from within an application to define a new macro or edit an existing one, a capability most useful for changing a few keystrokes in long or complex macros.

SuperKey offers a third way to define a macro; it allows the user to capture a block of information on the screen and insert it into a specified macro. This is useful for porting information from one application to another.

Because macro processors are memory-resident, care must be taken when loading them if other memory-resident software, such as RAM drives, spoolers, or desk utilities, is also used. The instructions should be read carefully, and, if possible, the macro processor should be tested with any other memory-resident software that will be used.

The market for macro processors is competitive. New products are being introduced rapidly, prices are dropping, and updated versions of older products are beginning to appear. Borland International has played a major role in deciding the fate of macro processors. Even before its low-priced SuperKey was released, competitors were rushing to match and exceed SuperKey's features. In many cases, the added features have nothing to do with macros; *keyboard enhancers* may be a more accurate term for these products.

A full range of macro processing programs is available. Several public domain programs and a few limited-function commercial products (none of which is reviewed here) are available through users' groups or bulletin board systems. This article deals primarily with the full-featured products that usually include other functions in addition to macros. (See table 1 for a feature-by-feature comparison of the commercial programs that are discussed here.) PC-DOS (2.0 or later) users have access to a simple (but limited and difficult to use) macro processor through the ANSI.SYS keyboard driver.

TABLE 1: *Comparison of Keyboard Macro Processors*

	KEYSWAP	KEYWORKS	PROKEY	RE/CALL	SMARTKEY	SUPERKEY
VERSION	3.04	1.0c	4.0	1.1a	5.0	1.0
REQUIREMENTS						
Minimum memory	64KB	128KB	96KB	64KB	64KB	128KB
Operating system	PC-DOS Any version	MS/PC-DOS 2.0 or above	MS/PC-DOS Any version	PC-DOS	MS/PC-DOS Any version	PC-DOS 2.0 or above
Other requirements	80 col. display needed for some features	PC-DOS recommended				
Copy-protected	No	No	No	No	No	No
DOCUMENTATION AND HELP FEATURES						
Format of manual	132 pages In binder	48 pages Booklet	158 pages In binder	94 pages Spiral-bound	105 pages Book	200 pages Book
On-line help	Limited to One screen	Context-sensitive	Limited	No	Limited	Context-sensitive
Quick-reference card	No	No	Yes	No	Yes	No
Sample macro set(s)	DOS, dBASE II, WordStar, SuperCalc	DOS, WordStar, 1-2-3, Multiplan	1-2-3, dBASE II, WordStar	1-2-3, Multiplan	WordStar, 1-2-3, dBASE II	1-2-3, WordStar, BASICA, DOS, VisiCalc
Level of feedback and prompts when creating a Macro	Poor	Excellent	Excellent	Not clear	Excellent	Excellent
DESCRIPTION AND LIMITATIONS						
Maximum macro buffer size	64KB	7K chars	50KB	40K chars	60K chars	64KB
Decrease in available memory when loaded (1KB buffer)	17,472 bytes	53,972 bytes	38,224 bytes	26,544 bytes	30,960 bytes	48,960 bytes
Regain memory if program is discontinued	Yes	Yes	No	N/A	Yes	Yes
User interface	Alt keys	Pop-up menus (optional)	Menus or Alt keys	Pop-up menus	Pop-up menus and 1-line descriptions	Pop-up menus
Suitable for use with a mouse (moving bar menus)	No	Yes	No	Yes	Yes	Yes
MACRO FEATURES						
User-definable command codes	Yes	Menu key only	Yes	No	Menu key only	Yes
Change command codes applications?	Yes	Yes	No	N/A	Yes	Yes
Define a macro via recall of keystrokes	No	No	No	No	Yes	No

A FREE MACRO PROCESSOR

Before examining the six commercial products to be reviewed here, a quick look at the built-in macro capabilities of PC-DOS may be of interest. The ANSI.SYS device driver gives the IBM PC extended screen and keyboard control; this control includes a limited macro capability. Perhaps the main point in ANSI.SYS's favor is its price: it is included at no extra cost with DOS versions 2.0 or later.

The major limitation of the ANSI.SYS macro capability is that it works only if the application uses DOS function calls for keyboard input. Most programs bypass DOS altogether and make keyboard function calls directly to BIOS. Therefore, macros that are developed under ANSI.SYS are somewhat limited in use. Another limitation of ANSI.SYS is the size of the storage buffer: 190 characters. Still another drawback is that ANSI.SYS is not easy to use, and IBM's documentation on the macro feature is not extensive.

Probably the best use for ANSI.SYS macros is for the automatic typing of DOS-level commands. Practically any key can be redefined under ANSI.SYS, including the extended-code keys (these include the function keys, cursor-control keys, and keys combined with Alt or Ctrl). The ANSI.SYS driver is loaded by including the following line in the CONFIG.SYS file:

```
DEVICE=ANSI.SYS
```

For specifics on how to use ANSI.SYS, refer to the DOS *Technical Reference* manual or Tech Notebook 14, "Defining Function Keys Using ANSI.SYS," Arthur A. Gleckler, *PC Tech Journal*, March, 1984, p. 77.

ANSI.SYS cannot be compared fairly to the other programs described below, but it is important to realize that limited macro capability is available to any user; users can try out simple macros before investing money in a program. ANSI.SYS may be the only macro processor many users will ever need.

KEYSWAP

KEYSWAP, from Rickerdata, includes most of the standard macro processor features, such as the ability to define macros within programs, to create macros that accept user input, and to customize the macro processor's control codes.

TABLE 1: *Continued*

	KEYSWAP	KEYWORKS	PROKEY	RE/CALL	SMARTKEY	SUPERKEY
Redefine extended characters via Alt key plus numeric keypad	Yes	Yes	No	Yes	Yes, same as SuperShift	Yes
Multiple character macro names	No	No	Yes	No	No	No
Turn all macros on/off from within a program	Yes	Yes	Yes	One by one	Yes	Yes
Use original meaning of a single key	Yes	Yes	Yes	Yes	Yes	Yes
Disable interpretation, but not playback, of macros from programs or DOS	Yes	Window macros only	No	No	Window macros only	Window macros only
Terminate a macro during execution	Yes	Yes	Yes	Yes	Yes	Yes
List all current macros	Yes	Yes	Yes	Yes	Yes	Yes
Macro menus in applications	Yes	Yes	No	Yes	Yes	Yes
Supports help window or display macros	Yes	Yes	No	Yes	Yes	Yes
Supports macro titles	No	Yes	Yes	No	Yes	Yes
Move macro to another key	Yes	With editor	No	Yes	Yes	Yes
Fixed and variable length user input	Yes	Yes	Yes	Yes	Yes	Yes
Format user input	Upper/lower-case only	No	No	No	No	Yes
Set toggle key states for subsequent keyboard input	Yes	No	No	No	No	Yes
Variable speed playback	Yes	No	Yes	Yes	Yes	Yes
Time delays at playback	Yes	Yes	Yes	Yes	Yes	Yes
Nested and recursive macros	Yes	Yes	Yes	Yes	Yes	Yes
Define new macro within a macro	Yes	Yes	Yes	No	Yes	Yes
Allows guarded macros	No	No	Yes	No	No	No
Requires verification before defining common keys	No	No	Yes	No	No	Yes
Requires verification before redefining an existing macro	No	Yes	Yes	Yes	Yes	Yes
Build macro from block of characters on screen	No	No	No	No	No	Yes
Hide macro creation from programs or DOS	Yes	With editor	With editor	Yes	Yes	With editor
Wait for disk to stop before macro playback	No	No	Yes	Yes	No	Yes
Macro editor included	Yes	Yes	Yes	Yes	Yes	Yes
Include sound in macros	Beep only	Beep only	No	No	Beep only	Programmable
Single-step mode for debugging macros	No	No	No	Yes	No	No

Additional functions, however, make the product worth investigating.

KEYSWAP holds the distinction of being the first macro processor available with an online macro editor. The editor is available at any time, even in the middle of an applications program; it is easy to use; and it allows the user to modify any macro currently in memory. KEYSWAP also allows the user to save an entire screen image and retrieve it later—a particularly useful feature in conjunction with the program's ability to make custom help screens.

Figure 1 is an example of a KEYSWAP macro file. The third macro in this figure illustrates a feature that is unique to KEYSWAP—the ability to define display-only text and, in the process, to provide user prompts. Characters between the Alt u commands are sent to the screen, but not to the current program (in the example's case, DOS). When the macro is invoked at the DOS level with Alt c, the macro displays:

```
A>
Enter filename to copy from:
```

With the exception of the word *copy*, this line is hidden from DOS. When the user enters the name of the source file, the macro is prompted to continue on the next line with:

```
Enter destination filename:
```

This line is entirely hidden from DOS. In effect, DOS sees only the word *copy*, followed by the two file names. The result is an improved copy command, complete with user prompts.

Another unique feature of KEYSWAP is its watchdog function. This allows a macro to be defined to execute after a specified period of time with no keyboard activity. This is useful in saving word processing files automatically when the user leaves the system for a while or in clearing the screen (after it has been saved) to prevent burning of the display's phosphors.

KEYSWAP also allows the user to invoke a macro that sends commands directly to the printer. This feature is invaluable for sending printer-control characters from within any application.

KEYSWAP's documentation, complete with an IBM-style binder, seems almost overdone. Rickerdata obviously went to considerable expense in producing the documentation, but it contains several

TABLE 1: *Continued*

	KEYSWAP	KEYWORKS	PROKEY	RE/CALL	SMARTKEY	SUPERKEY
FILE HANDLING						
Read or write macro definition files from within programs	No	Yes	Yes	Yes	Yes	Yes
Macro file format	ASCII	ASCII	ASCII	Binary	ASCII	ASCII
Reads ProKey files	No	Conversion program included	Yes	No	Conversion program included	Yes
Autoexecution of a macro when loading a macro file	Only when loaded from DOS	Only when loaded from DOS	Yes	Yes	Only when loaded from DOS	Yes
Merge sets of macros in memory	Yes	Yes	Yes	Yes	Yes	Yes
Requires user verification to overwrite macro files	No	Yes	No	No	Yes	Yes
Requires user verification before overwriting unsaved macros in memory	No	Yes	No	No	No	Yes
Recovery from disk errors	Uses DOS error messages	Good	Good	Poor	Good	Good
ADDITIONAL FEATURES AND UTILITIES						
Automatic date/time macro	Yes	No	Date only	No	No	Yes
One-finger mode for multiple key characters	No	No	Yes	Yes	Yes	Yes
Execute DOS commands from an application	No	Yes	No	No	Yes	No
DOS command stack for recall/editing	No	No	No	No	No	Yes
Automatic macro execution if no keyboard activity in specified time	Yes	No	No	No	No	No
Expanded type-ahead buffer	90 chars	No	Selectable to 250 chars	No	128 chars	128 chars
File encryption	No	Yes	No	No	Yes	2 types
Send control codes to printer from an application	Yes	Yes	No	No	No	No
Auto screen blanking	With macro	Yes	Yes	No	Yes	Yes
Keyboard click on/off	No	No	No	No	No	Yes
Keyboard lock	No	No	No	No	Yes	Yes
Give status of toggle key states	Audible	No	No	No	No	No
Dvorak setup included	Yes	No	Yes	No	Yes	Yes

A full range of macro processors is available; the products summarized in this table are full-featured packages that usually include additional functions besides macros. Keyboard enhancers may be a more appropriate term for these products.

errors, and some of the complex functions lack detailed explanations and examples. A quick-reference card listing the commands and their syntax would have been helpful.

The latest version of KEYSWAP (3.04) includes two new features: windowing and keyboard filtering (preliminary editing of input). The documentation for these features is a poorly written addendum to the user manual, and getting these features to function properly is difficult. KEYSWAP even includes a READ.ME file on its distribution disk with a description of windowing followed by the warning, "Caveat user!"

The program's on-screen tutorial is a series of macros that can be selected by the user via function keys. These macros are then displayed on the screen using KEYSWAP's slow macro playback mode. The result is an informative but tedious tutorial.

Also included in the READ.ME file is technical information on how to invoke KEYSWAP macros from a program. None of the other programs reviewed here supplies any such technical information for programmers.

One drawback of KEYSWAP is its lack of visual feedback. When a new macro is defined in record mode, KEYSWAP emits a short auditory signal and the next key pressed will be redefined. Unlike most other programs, KEYSWAP does not display the name of the key being defined, and it is easy to inadvertently redefine the wrong key. Visual feedback helps prevent errors and gives the program a more solid feel.

Another of KEYSWAP's weaknesses is that saving and reading macro files can be done only at the DOS level. This is acceptable for many applications, but in some cases being able to read and write macro files from an applications program is useful, because the macros can be tested and modified easily.

KEYSWAP's commands all use the Alt key, and users may discover that some macro names that they want to use are reserved for KEYSWAP commands. Furthermore, many of the commands are not particularly mnemonic. For example, pressing Alt q displays the date, and pressing Alt r displays the time. These problems can be overcome, however, by calling up the feature that enables the user completely to customize KEYSWAP's commands.

A major advantage of KEYSWAP is that it uses the least memory of any program reviewed here—only about 18KB in a minimal configuration.

To summarize, KEYSWAP is a capable macro processor that includes several unique features. Unfortunately, its newest features are so poorly documented that they are practically unusable. KEYSWAP seems less advanced than the other programs reviewed here. Its user interface is not as intuitive as most, and the new functions are not well integrated into the program.

KEYWORKS

Alpha Software's Keyworks has the best user interface of any of the programs reviewed. It is easy to use, and its commands are available through an intuitive set of pop-up menus. Commands can be accessed via a moving bar that is controlled by the cursor-control keys. Alternatively, the first character of a menu item can be typed to select an item. The program works nicely with a mouse—a custom mouse program can be set up using Mouse System's Designer Pop-up Menu software.

Perhaps Keyworks' strongest feature is its menu capability. Custom pop-up menus of macros can be set up in a professional-looking manner. Furthermore, unlike the menu options in the other macro processors, Keyworks menus are easy to establish. The program uses a template that pops up when the Define Menu option is selected. A similar template is available for desiging pop-up help windows.

The Keyworks program provides context-sensitive help and excellent prompts. For experienced users, most

Perhaps the strongest feature of Keyworks is its menu capability. The user can set up custom pop-up menus of macros in a professional-looking manner.

of these prompts can be eliminated if desired. In addition, the program can be set up to work like WordStar: the menus can be made to appear only if the user hesitates for a predetermined length of time. With this system, an experienced user can completely avoid the menus, but the novice still is able to get assistance if necessary.

Keyworks obviously was influenced by Borland International's SuperKey. For example, Keyworks offers memory-resident file encryption and the ability to call SideKick from a macro—two features pioneered by SuperKey. Keyworks also provides some useful functions that are not available with SuperKey. Several DOS functions (including the FORMAT command) can be accessed from within any applications program. Users who have lost a file because of insufficient disk space and have no formatted blank disks on hand will appreciate this feature. In addition, when loading a file of macros, the user can select the desired file name using another moving bar menu. Alpha also distributes an excellent on-screen tutorial with Keyworks.

One nice option is Keyworks' ability to send control codes to the printer from within any program. Type size, type style, margins, and other factors can be changed without exiting to DOS. This feature is not documented in the manual, but it is explained in the READ.ME file on the distribution disk.

Keyworks also includes a program to translate ProKey macro files into its format. (See figure 2 for a sample macro file from Keyworks.)

FIGURE 1: *KEYSWAP Macro Definition File*

```
MACRO
<alt=><f1>dir a:/w<cr>
<alt->
MACRO
<alt=><altf1>dir <^]> <^]>:/w<cr>
<alt->
MACRO
<alt=><altc><altu><cr>
Enter filename to <altu>copy <altu>from:
<altu><^-><^-><altu><cr>
Enter destination filename:<altu> <^-><^-><cr>
<alt->
```

F1 gives a wide directory of drive A:. Alt f1 waits for user input of drive specification (fixed-length field). Alt c simplifies copying files by prompting the user for source and destination file names (variable-length fields). Characters embedded between the pairs of Alt u commands will not be interpreted by DOS.

FIGURE 2: *Keyworks Macro Definition File*

```
{{AltA}<Dir of a:          >dir a:{Enter}
{{AltB}<Dir of b:          >dir b:{Enter}
{{AltM}<Directory Menu>{Menu}{Title}DIRECTORY MENU\{IdOff}
{Exit}{Esc}{Select}{AltA}{AltB}{Menu}
```

Alt A and Alt B are used for directories of drives A: and B: respectively. Alt M is a menu macro that displays the titles of the two defined macros (the text between the angle brackets) and that allows selection via a moving bar in the menu. Alt M was defined using Keyworks' menu template.

The price the user pays for these features is memory. Keyworks takes up more memory (52KB) than any of the other programs reviewed. With the large amounts of memory installed in most users' systems, however, this is not a serious problem.

The documentation consists of a 46-page booklet that is well-designed and clearly written. It should be updated, however, to include the program's new features. A new user should have no trouble learning how to use Keyworks, including its advanced functions of menus and windows.

Keyworks lacks some of the features available in many of the other products. It has a relatively small maximum macro buffer, does not allow variable-speed macro playback, and will not interrupt macro execution while the disk is being accessed. However, Keyworks' other features, its ease of use, and its on-line help, make it a strong product.

PROKEY

To many people, Rosesoft's ProKey is synonymous with keyboard macros. It has outsold all of its competitors, and Rosesoft boasts it has more than 50,000 ProKey users. The product's predominance in the marketplace is evidenced by the fact that three other programs are able to read (or translate) macro files written by ProKey.

Version 4.0 of ProKey includes several new features: online editing, longer macro names (up to eight characters), listing of current macros, and a guard feature that keeps selected macros in memory until the system is turned off. Many users will be pleased that this version is completely memory-resident. ProKey can now read and save macro files without going to DOS.

Surprisingly, ProKey's latest version still does not supply any on-line help. Available help is limited to a simple menu that lists the commands and then refers the user to the manual for more information. In addition, ProKey's menu is not the moving-bar type; in order to select a command from a ProKey menu, the user must type the first letter of the command. While this is easy to do, it is not as intuitive as the moving-bar type of menu; it also precludes the use of a mouse with ProKey.

FIGURE 3: *ProKey Macro Definition File*

```
<begdef><f1>dir a:/w<enter>
<enddef>
*
<begdef><altf1>dir <ffld>.<ffld>:/w<enter>
<enddef>
*
<begdef><altc>copy a:<bedgef><altf><enddef> b:<enter>
erase a:<altf><enter>
<altc><enddef>
*
```

F1 gives a wide directory for drive A:. Alt f1 waits for user input of drive specification (fixed-length field). Alt c copies files from drive A: to drive B:, then deletes the file from drive A:. It repeats until stopped with ProKey's Ctrl Esc command. Note the nesting of another macro and use of recursion.

One of ProKey's strengths is its simplicity. A macro can be defined at any time simply by pressing Alt =. A message appears on the screen asking which key should be defined, and other prompts guide the user throughout the keystroke recording. If the user indicates he wants to define a common key, such as *a* or Esc, ProKey requires verification. This is an excellent precaution and is found in only one other program, SuperKey. The Alt – combination ends the macro definition. (See figure 3 for a sample ProKey definition file.)

ProKey's menu, in addition to supplying help to the user, can also be used to create and modify macro files. From the menu, ProKey macro files can be read from or written to memory, macro files can be modified with the built-in editor, or special functions can be inserted into a macro.

Any of ProKey's commands can be changed to avoid conflicts with other software. Unfortunately, this change must be made at the time ProKey is loaded. If the user neglects to make the change then and inadvertently starts a program that shares one or more of ProKey's commands, he cannot change any of those commands while the program is running.

ProKey originated the "one-finger mode," which enables key combinations such as Ctrl-C to be entered sequentially. This mode is effective with the Ctrl, Alt, and Shift keys. ProKey also was first with a special mode that suspends macro output while the disk drive is spinning. Some programs are designed to clear the keyboard buffer following disk access, and would therefore throw away any macro expansions sent during this time. ProKey's disk wait mode prevents this from happening. In addition, ProKey allows the user to define and specify alternate keyboard layouts, including the Dvorak system.

ProKey was the first macro processor program for the IBM PC, and Rosesoft originated many of the macro features that are now standard in many other programs. ProKey's no-frills nature is appealing to many users. Although ProKey lags far behind some of the other programs in features, current ProKey aficionados will appreciate the extensions in version 4.0. Another advantage in upgrading to the new version is that ProKey users' finely tuned macro files are guaranteed to be perfectly compatible with the new version—this advantage cannot be said for the other programs described here.

RE/CALL

RE/Call was developed by Yes Software, a Canadian firm. It has most of the standard features expected in a new product, but it is difficult to learn and use. The user can easily get stuck in one of RE/Call's many modes and not be able to continue without consulting the manual. The program badly needs a context-sensitive help feature; unfortunately, it offers no on-line help.

Furthermore, the version of RE/Call that was reviewed was copy-protected. It can be copied to a hard disk, but the key disk must be inserted every time the program is loaded. This feature alone is enough to make most users avoid the program at all costs. Fortunately, a new version of the software that is *not* copy-protected will be released by the time this review is printed. Registered users will receive the unprotected version free of charge.

RE/Call uses a separate status line when recording or examining a macro. Macro definitions are echoed in this line as they are recorded, causing

screen flicker, a jumbled screen, and much confusion. The main pop-up menu is invoked by typing Alt [. This key combination, as well as the other control commands, cannot be changed.

One point in RE/Call's favor is a unique feature known as single-step mode. This feature, which allows the user to play back a macro one character at a time, could be useful when trying to debug complex macros.

RE/Call's macros are stored in non-ASCII files. Therefore, any editing must be done in RE/Call's built-in editor. This editor displays one line at a time for editing, making it somewhat inefficient. The program supports macro menus, which can pop up in any application. Making them do so is easier said than done, however, and demands constant referral to the manual.

The version of RE/Call that was tested for this review contains a problem. If the user attempts to save a macro file on a write-protected disk, the program prompts him to "Retry or Ignore." The "Ignore" option does not work, and the only way out is to remove the write-protect tab.

SMARTKEY

Although ProKey was the first macro processor available for the IBM PC, SmartKey was the first available for microcomputers. First released for CP/M machines in 1979, SmartKey is now available for a wide range of systems based on both MS-DOS and CP/M. Version 5.0 comes bundled with a copy of SmartPrint–a program that allows the user to insert special printer-control codes into any document, including spreadsheets. SmartPrint and SmartKey work together, and the combination provides a useful array of features.

In addition, the distribution disk contains other programs that provide automatic screen blanking, file encryption, "one-finger" macro invocation, alternative keyboard layouts, and the ability to translate ProKey files. Because these functions are provided in separate programs, the user can use only the programs he needs, and is able to save memory for other uses.

SmartKey can be invoked at any time after it is loaded by pressing the key defined as the "smart key." The default smart key is the plus key on the numeric pad, but it can be changed at any time, even within another program. Pressing the smart key brings up a menu, with single-line descriptions similar in format to the 1-2-3 menu system.

A unique feature of SmartKey is its super-shift character, which significantly increases the number of macros that can be defined by allowing the user to specify a key to function in much the same way as the Shift, Alt, and Ctrl keys do. The default super shift is the minus key on the numeric keypad, but it too can be changed.

The super-shift feature simplifies the process of defining mnemonic macros. It is particularly useful in programs, such as WordStar, that use many Ctrl commands and consequently leave fewer key combinations that can be redefined as macros. Actually, the super shift mode translates to the extended character set that is accessible by typing Alt with a three-digit number on the numeric keypad. For example, the numeric keypad combination of Alt 198 translates to SS-f—that is, "super shift f."

The defaults can be changed by running an installation program from DOS. This program also allows the buffer size, the window size and colors, macro playback speed, and other functions to be customized.

SmartKey is easy to use, and its menu system is designed so the most common choices are easy to get to. DOS commands can be executed from within an application. SmartKey can be uninstalled, releasing the memory for another application, and SmartKey is the only product that indicates how much buffer space remains for macros.

Another unique feature of SmartKey is its ability to define a macro after the fact. When SmartKey is active, the user can recapture as many as the last 64 keystrokes and store them in a macro. A sample macro definition file is shown in figure 4.

Other SmartKey features include a keyboard lock function and the ability to list all macros by key, title, or full definition. The product also allows the construction of help windows and macro menus, though it is not easy to use either of these features, and the manual's assistance is required. These are the only functions in SmartKey, however, that are not immediately intuitive.

SmartKey has evolved into a competitive product. It is efficient in memory usage, easy to use, and can be customized in several ways. Software Research Technology has taken a different approach in including nonmacro features, such as screen blanking, file encryption, and printer control. These are contained in separate nonresident programs and therefore do not take up memory if they are not needed. Considering all the extra programs bundled with SmartKey, at $49.95, it is perhaps the best buy on the market.

FIGURE 4: *SmartKey Macro Definition File*

```
<[F1]>      <dir a:/w[Enter]>
<[Alt D]>   <dir <K1>:*.bak[Enter]>
<[Alt T]>   < <Ct><T3>[Bksp][Bksp][Bksp][Bksp][Bksp][Bksp][Bksp][Bksp]>
```

F1 gives a wide directory of drive A:. Alt D waits for user input of drive specification (fixed-length field) and gives a directory of all *. BAK files on that drive. Alt T displays the time for a period of three seconds.

FIGURE 5: *SuperKey Macro Definition File*

```
<BEGDEF><AltD>dir <FFLD> <FFLD>:*.com<ENTER>
<ENDDEF>
<BEGDEF><AltT><CMD>FB05000 00450 00100<CMD><CMD>FT<CMD>
<ENDDEF>
<BEGDEF><AltS><SIDEKICK>n<ENDDEF>
```

Alt D illustrates a fixed-field macro. It waits for a drive letter, then gives a directory of all .COM files. Alt T sounds a tone that begins at 5,000 Hz, ends at 450 Hz, and lasts for one second. Immediately following the tone it displays the time. Alt S calls SideKick through a special SuperKey command and enters its notepad function.

SUPERKEY

Borland International's entry into the macro processor market is an impressive product. In addition to the standard macro processing functions, SuperKey includes two types of file encryption (see "Encryption Software," Victor Mansfield, *PC Tech Journal,* June 1985, p. 162, for a review of the encryption utilities), a keyboard-lock feature with limited usefulness, a screen-blanking function, and optional keyboard click. Borland clearly is attempting to lure the large base of ProKey users; SuperKey can read ProKey files directly—that is, no translation utility is needed.

SuperKey is easy to learn and use, and context-sensitive help is available at any time. It is a menu-based system, but the user can avoid the menus by using its other commands. SuperKey works well with a mouse and, like Borland's SideKick, can be completely customized with user-selected colors.

See figure 5 for an example of a SuperKey macro file.

All of SuperKey's features are available at any time. This includes changing commands, changing defaults, and even file encryption. Depending on the user's need for these features, this could be an advantage or a drawback because the memory is used whether or not the features are needed.

SuperKey was designed to call up SideKick, but only SideKick version 1.5 or later will work. Attempting to call up an earlier version results in the display of an advertisement urging the user to upgrade (for about $20).

SuperKey allows the user to include a wide variety of sounds in his macros. The starting and ending frequency, as well as the length of the time, can be specified. This feature can be handy in getting someone's attention or in making a word processor sound like an arcade game. A keyboard click can also be specified. The sound is barely audible and is useful when a key is held down and repeats.

SuperKey includes several more useful features. One is its cut-and-paste option that allows the user to include any information on his screen in a macro. This could be useful in porting information from one application to another. Finally, like ProKey, SuperKey requires user verification before it will define one of the common keys.

Another nice feature, unrelated to macros, is SuperKey's ability to recall DOS commands. The program maintains a number of previously executed DOS commands in an easily accessible stack. A previous command can be recalled and even edited by adding, deleting, or changing any characters. This is particularly helpful for recalling long command strings.

SuperKey has a unique ability to format fill-in-the-blank information in a macro. All macro processors allow variable- or fixed-length fields. When the macro plays backs, it stops to allow this variable information to be entered. SuperKey takes this idea one step further and provides a formatting function using an entry mask. For example, the macro could be set up to accept only

mbp COBOL for your IBM/PC

Now, more convenient... 4 (exclusive) ways.

Now, the mbp COBOL Compiler offers unrivaled convenience to go with its unmatched performance.

Here are the convenience features you've wished for:

1) an enhanced Screen Management System with program-controlled video attributes and color; **2)** support for PATH & sub-directories; **3)** DOS command execution from within a COBOL program; **4)** 'permanent' DEFAULT modification.

The new mbp Compiler has them all! And they're exclusives: you get them *only* with mbp.

GIBSON MIX Benchmark Results
Calculated S-Profile
(Representative COBOL statement mix)
Execution time ratio

mbp* COBOL	Level II* COBOL	R-M* COBOL	Microsoft* COBOL
1.00	4.08	5.98	6.18

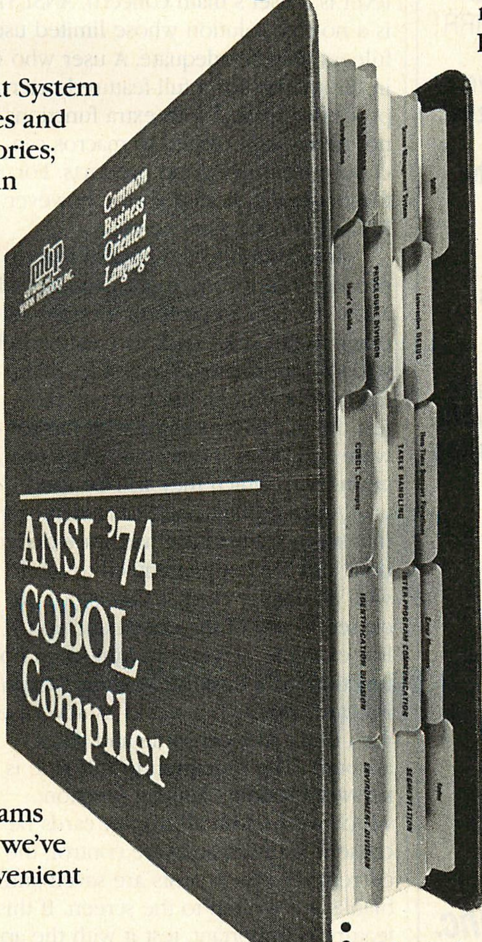
*128K system with hard disk required. IBM/PC & /AT are IBM TMs; Netware is a Novell, Inc. TM; Level II is a Micro Focus TM; R-M is a Ryan-McFarland TM; Microsoft is a Microsoft TM.

Plus, it's 4 times faster.

Because the mbp COBOL Compiler generates native machine language object code, it executes programs *at least* 4 times faster (see chart). Now, we've made that performance even more convenient to use.

The complete COBOL.

An Interactive Symbolic Debug Package included standard; Multi-keyed ISAM structure; SORT & CHAIN; GSA certification to ANSI '74 Level II; IBM/PC-AT and TI Professional compatibility; with mbp, you get it all. Optional: IBM-PC Network and Novell interfaces.



mbp COBOL: the choice of professionals.

It's no surprise more and more companies like Bechtel, Bank of America, Chase, Citicorp, Connecticut Mutual, Hughes Aircraft, McDonnell-Douglass, and Price-Waterhouse choose mbp COBOL.

Make it your choice, too. Just send the coupon, or call, for complete information. Today.

Visit us at COMDEX, Booth 4658

mbp COBOL. $1000

Please send complete mbp COBOL information to:

NAME ____________________

COMPANY ____________________

ADDRESS ____________________

CITY/STATE/ZIP ____________________

____

PHONE ____________________

mbp Software & Systems Technology, Inc.
1131 Harbor Bay Parkway, Suite 260
Alameda, CA 94501
Phone 415/769-5333

mbp®

CIRCLE NO. 172 ON READER SERVICE CARD

numeric input, only uppercase alpha input, only lowercase input, etc. If only uppercase input is accepted, SuperKey will convert lowercase characters to uppercase. Similarly, fields can be set up to be justified to the right or left or centered, and numeric input can be formatted in a manner similar to that used by BASIC's PRINT USING statement.

Still another feature available with SuperKey is display–only macros. These can be used for help windows or menus within applications. SuperKey's editor cannot edit display macros. This is unfortunate, because display macros are precisely those macros that require debugging. A user who is developing a complex display macro should forget SuperKey's online editor and load his own text editor instead.

Of the six products reviewed here, SuperKey has the most features. It is easy to use, offers context-sensitive help, has excellent documentation, and is able to work together with Borland's popular utility, SideKick.

WHICH TO CHOOSE?

If saving a few keystrokes at the DOS level is a user's main concern, ANSI.SYS is a no-cost solution whose limited usefulness may be adequate. A user who is in the market for a full-featured macro processor loaded with extra functions not necessarily related to macros can choose from five good products. For the reasons indicated above, however, RE/Call is not recommended.

In addition to the normal macro processor functions, these products will allow the user to perform a variety of other useful tasks, such as creating help windows or menus. If possible, potential buyers should, prior to purchase, test the macro processor with the software and other memory-resident programs with which they will be used.

An important point to remember is that some features are hardware-dependent and are not guaranteed to work with all display cards. (This is not discussed in any of the documentation.) For example, screen blanking does not function on a Paradise Modular Graphics card, although it works as advertised with IBM's monochrome and color adapters. The exception to this rule is KEYSWAP's screen-blanking function, which works with all display cards because it is under software control; the current screen contents are saved and blanks are written to the screen. If this feature is important, test it with the appropriate hardware before buying.

Keyworks' macro menu and window features are elegant and, unlike all similar features offered by the other programs, can be used without referring to the documentation. SuperKey has file encryption and the ability to recall and edit DOS commands, and its ability to include sound within macros is attractive. It has a tremendous number of functions and will appeal to SideKick users and others who admire Borland's corporate strategy. The new version of ProKey is attractive because of its simple elegance and the large number of current users. SmartKey has a great list of features and is an exceptional value. Finally, KEYSWAP, while lacking in features that improve the ease of use, provides many functions and uses minimal memory.

KEYSWAP: $69.95
Rickerdata, Inc.
P.O. Box 998
Melrose, MA 02176
617/662-0856
CIRCLE 301 ON READER SERVICE CARD

Keyworks: $89.95
Alpha Software Corp.
30 B Street
Burlington, MA 01803
617/229-2924
CIRCLE 302 ON READER SERVICE CARD

ProKey: $129.95
RoseSoft, Inc.
4710 University Way NE
Suite 601
Seattle, WA 98105
206/524-2350
CIRCLE 303 ON READER SERVICE CARD

RE/Call: $135.00
Yes Software, Inc.
#309-10991 Shellbridge Way
Richmond, BC, Canada V6X3C6
800/663-0373
CIRCLE 304 ON READER SERVICE CARD

SmartKey: $49.95
Software Research Technology
3757 Wilshire Blvd.
Suite 211
Los Angeles, CA 90010
213/384-5430
CIRCLE 305 ON READER SERVICE CARD

SuperKey: $69.95
Borland International
4311 Scotts Valley Drive
Scotts Valley, CA 95066
408/438-8400
CIRCLE 306 ON READER SERVICE CARD

John Walkenbach has a Ph.D. in experimental psychology and is a senior vice president at a financial institution in California.

Everything You Always Wanted To Know About

APPLICATION DEVELOPMENT DATABASES...*

DATA

Just What Is DataFlex, Anyway?

The easiest thing to say is that DataFlex is an application development system *like* d-BASE. However, that wouldn't be a fair statement about either product because DataFlex is not only faster, more powerful and easier to use than d-BASE, but also supports TRUE multi-user transaction processing with complete data integrity. Applications developed with DataFlex can also be run UNCHANGED on a wide selection of 8 and 16-bit operating systems and LANs.

More Powerful and Easier To Use?

Absolutely. DataFlex uses "image formatting" to quickly and efficiently develop input screens and report formats for your application. All you have to do is make an "image" of your screen or form using any ASCII text editor. AUTODEF, DataFlex's file definition utility, automatically generates an error-free, ready to compile data entry program, creates the necessary data and key index files, and makes a data dictionary entry describing each field, its length, type and format. With DataFlex, there's no need for you to go back and provide tedious definitions of the length, format and data type of your windows. And while we're on the subject of text editors, you should know that DataFlex is also available with a full function word processing option that's operationally equivalent to and compatible with MicroPro's WordStar + Mailmerge.

What About A Procedural Language?

DataFlex has a powerful procedural language that combines the best features of Pascal, BASIC and RPG. It has over 125 commands in the following categories:

Argument Processing	Indicators
Console I/O	Key Procedures
Control	Multi-User Functions
Database Commands	Reporting
Definition	Sequential I/O
Data Entry	String Operations
Forms Processing	Structural Control
	System Commands

DataFlex allows programs of up to 2,750 command lines incorporating up to 255 screen images! This enables you to design systems far more complex and sophisticated than possible with some other database programs. For many applications, however, knowledge of the procedural language is not necessary since DataFlex's powerful AUTODEF and QUERY functions automatically generate the source code for handling data entry and reporting. Data checking and formatting commands and error traps can easily be added to the source code before compiling so your applications will run smoothly with minimal possibility of operator error.

How Can DataFlex Be So Fast?

DataFlex utilizes a multi-keyed B+ ISAM structure which updates indexes on-line each time data is entered, deleted or edited. Since all data is instantly available for recall, time consuming key sorts and batch index reorganizations are not necessary. With DataFlex, you'll never again have to wonder whether or not your computer is really working or "hung-up" somewhere in the middle of a sort. DataFlex reports appear on your screen or printer as quickly as the data can be read from your disk. Data input is also speeded by DataFlex's FlexKeys™: single keystroke commands that perform record finding, saving and editing functions. There's even a HELP FlexKey that can summon instructions or explanations pertinent to your application.

Will DataFlex Exchange Data With Other Programs So I Don't Have To Start All Over From Scratch?

Yes! Import and export of data is a snap. DataFlex can read or output either comma or carriage return/linefeed delimited files. Conversion of d-BASE data files is described in detail in our comprehensive user's manual.

The Application Development Database For Single-User, Multi-User And Local Area Networks

FLEX™

What's DataFlex Have For Those Of Us With Heavy Reporting Requirements?

Output for reports, labels, preprinted forms and files is handled by a powerful DataFlex macro command called REPORT. It consists of an integrated set of selectable predefined output routines that can be chosen as needed. These routines are used to "fill-in" an output "image" and then send it to the device of your choice. You can write the report command file yourself or it can be automatically generated by DataFlex's QUERY function. Even complex multi-file reports can be generated through QUERY. All you do, using the arrow keys or a mouse, is "point-and-shoot" at the data you want to see! QUERY then automatically writes error-free source code and allows you to save it as an ASCII file that you can then customize, compile and run. Output can be sent directly to your printer or CRT, or saved as a comma or carriage return/linefeed delimited ASCII file for later use by DataFlex or some other program. The speed with which QUERY performs its source code generation function is something that you have to see for yourself to fully appreciate.

SUPPORTED OPERATING SYSTEMS AND NETWORKS

- IBM PCDOS 1.x, 2.x, 3.x
- MSDOS 1.x, 2.x, 3.x
- IBM "AT" XENIX
- IBM PC NETWORK
- CP/M, CP/M-86
- Concurrent CP/M-86
- Concurrent DOS
- MP/M-86
- Novell NetWare
- TurboDos
- Corvus w/IBM PC
- Molecular N/Star
- TeleVideo Infoshare
- 3Com EtherShare
- PC-Net
- NorthStar Dimension
- Action DPC/OS
- DMS Hi-Net
- Alloy Engineering RTNX

SPECIFICATIONS

Environment:
- 8080, Z80, 8088, 8086, 80186, 80286

Requirements:
- 52K TPA (8-bit)
- 256K TPA (16-bit)
- CRT w/cursor addressing
- 600K disk storage

Capabilities:
- 255 Database files
- No limit on number of open files (16-bit)
- 9 6-segment indexes per file (16-bit)
- 16K Bytes per record
- 255 Fields per record
- 16,777,215 Records/file
- 2,750 Program lines per configuration
- 255 Screens per configuration
- 2,900 Windows per configuration
- 32,000 Variables
- 127 Indicators
- 9 Global break point levels
- 18 Terminal independent function keys

Does DataFlex Work On MULTI-USER Systems and LANs?

Yes! DataFlex, unlike most other products, supports TRUE multi-user processing. With DataFlex, no user is ever locked out of a file or denied access to a record. Every user can, at any time, access, read and even change any record in the database while maintaining absolute data integrity!

How Easy Is It For The "End User" To Deal With DataFlex Applications?

DataFlex includes an elegant menu system which totally insulates the end-user from the computer's operating system. Each menu screen supports up to nine prompted actions each, including chaining to "sub-menus" and DataFlex programs, and the execution of system commands and other programs. A pre-programmed "help-screen" is included to provide operator assistance on selecting items from the menu. Password security can be established for each menu action to prohibit unauthorized file access, and the passwords (or even the entire menu) can easily be changed at any time by programmers with access to DataFlex's MENUDEF utility.

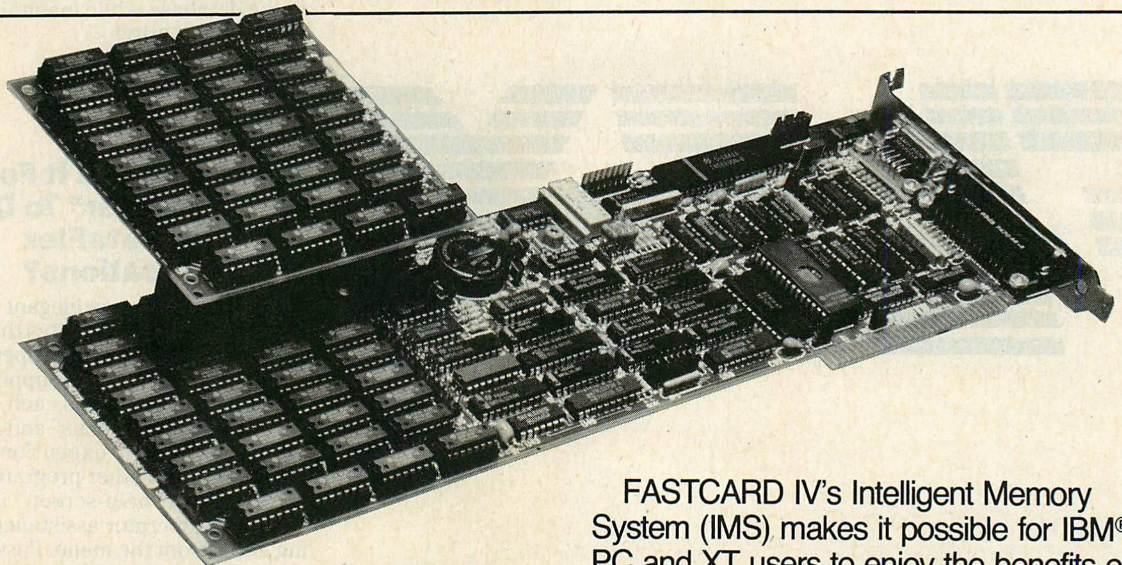
With Some Programs, I've Had Performance Go Right Down The Tube When There's More Than A Thousand Or So Records. How's DataFlex Perform With Large Databases?

Large, complex databases are exactly what DataFlex was designed for, and its performance in this environment is impressive. In benchmark tests on a Wang PC with a 36,000 record database of 128 byte records, DataFlex was able to find a record via a 41 byte key and display it to the screen in .8 seconds! This high level of performance extends to multi-file operations as well, where the PC version of DataFlex puts no limit on the number of open database files. As many as 255 database files can be maintained by DataFlex, the size of each limited only by your operating system and DataFlex's 16.7 million record per file "limit."

DataAccess Corp., 8525 S.W. 129th Terrace, Miami, FL 33156 (305) 238-0012 TELEX 469021 DATA ACCESS CI

FASTCARD IV

TWO MEGABYTES FOR YOUR PC—AND MORE!



The power of the Lotus® Expanded Memory Specification is now available in an intelligent, multifunction expansion board—THESYS' FASTCARD IV™.

FASTCARD IV's Intelligent Memory System (IMS) makes it possible for IBM® PC and XT users to enjoy the benefits of up to two megabytes of expanded memory with *all* the software in their libraries—*not just programs written especially for expanded memory.*

THESYS' IMS features include:

- **Disk Enhancement**—Frequently accessed disk information is stored in system RAM or expanded memory where it may be rapidly retrieved.
- **Printer Enhancement**—IMS allocates memory to serve as a holding area for information waiting to be printed.
- **Automatic Memory Allocation**—IMS monitors the computer's operations and allocates the system and expanded memory as needed for disk enhancement, printer enhancement, or other applications.

In addition, FASTCARD IV offers **Data Security** features vital to PC users:

- **Custom Password Security** enables the user to define a password that must be entered *before the computer will boot.*
- **Built-in Diagnostics** and **Auto Fault Tolerance** identify faulty memory and bypass it until it can be replaced.

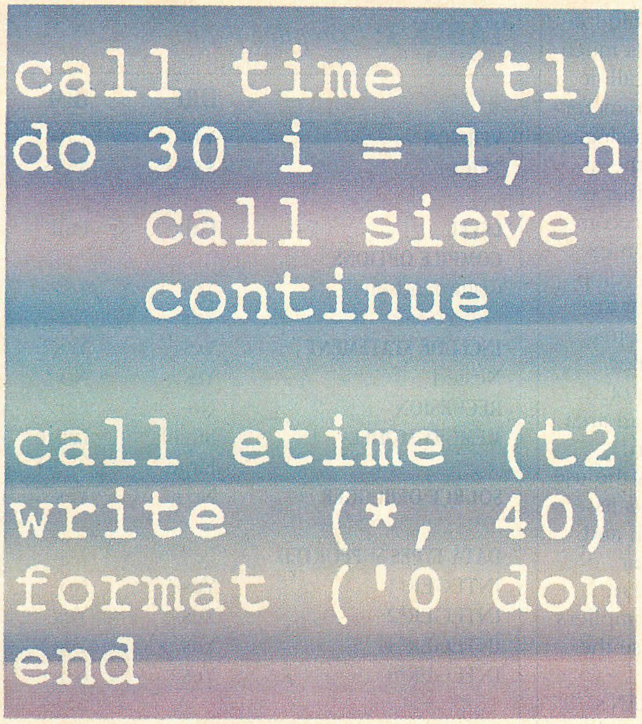
FASTCARD IV is **Multifunction**, featuring:

- **Serial** and **Parallel Ports** and **Game Port Interface**
- **Battery-Powered Clock/Calender**
- Easy-to-use **Installation Software.**

THESYS
MEMORY PRODUCTS CORPORATION

For ordering information, contact:
THESYS Memory Products Corporation
7345 East Acoma Drive
Scottsdale, AZ 85260
(800)-327-8345
(602)-991-7356

CIRCLE NO. 188 ON READER SERVICE CARD



FORTRAN Options

ALAN HOWARD

Four compilers let programmers choose among the finer points in a FORTRAN implementation.

FORTRAN has been used on a wide range of computers since its introduction in the mid 1950s. Issued originally for the IBM 704 in 1957, it was the first language to be standardized by the American National Standards Institute (ANSI), then called the American Standards Association. The first standard defined two languages, basic FORTRAN and FORTRAN. The current standard, ANSI X3.9-1978, is commonly known as FORTRAN 77. This article reviews FORTRAN 77 compilers for the PC from four companies: Digital Research, Inc., IBM, Lahey Computer Systems, and Microsoft. (Examined here are the versions that were available at the time the article was written; since that time, newer versions have been released.) The features of each compiler are listed in table 1.

Even with the early standardization, a major problem with FORTRAN compilers has been the addition of nonstandard features by suppliers. Prior to the advent of microcomputers, compilers normally were supplied by the manufacturers of computer hardware. It was advantageous for them to include compiler features offered by competitors to make the conversion to new hardware relatively easy. Yet they often offered extensions that made conversion to other hardware more difficult, as well. UNIVAC, for example, supplied a program called SIFT to aid in the conversion from IBM FORTRAN IV to UNIVAC FORTRAN V. In an environment where the average life of a mainframe computer is three to four years, the cost of converting hundreds of programs to run on a new computer is a significant factor in the choice of new hardware.

The addition of features in itself is not a problem. The difficulty arises when each supplier implements a feature differently, making its use a problem if a program has to be converted for another compiler or computer.

For example, each of the compilers reviewed provides an **include** command

ILLUSTRATION • MARK TAUSS

to insert the text of an external file into the current file. This command is much like a compile-time subroutine call; it permits common blocks and parameter statements to be maintained in a single file. When a change is made to the include file, only the source files that refer to the include files must be recompiled. The include statement is an extremely useful feature for the development of any but the smallest software systems, yet the syntax for this command is different in each compiler.

Another example is the use of lowercase letters in identifiers (an area not covered by the ANSI standard). Each of these compilers permits the use of lowercase letters, but each has its own rules about what they mean and where they may be used. To their credit, several of the extensions are common to earlier FORTRAN compilers and are included in order to ease the conversion to FORTRAN 77.

Portability is one of FORTRAN's strongest advantages; the concern about transporting software is not trivial.

TABLE 1: *Compiler Features*

	DRI	IBM	LAHEY	MICROSOFT
VERSION	4.0	1.0	1.1	3.2/3.3
PRICE	$350	$595	$477	$350
LEVEL	Full	Full	Full	Subset
COMPILE OPTIONS				
Command line	Yes	Yes	Yes	No
Source file	Yes	No	Yes	Yes
INCLUDE STATEMENT	Yes	Yes	Yes	Yes
Nested	Yes	No	Yes	Yes
RECURSION	No	No	Yes	No
MEMORY MODEL	Small, large	Large	Large	Large
SOURCE DEBUGGER	No	Yes	Yes	Yes
DATA TYPES SUPPORTED				
INTEGER*1	Yes	No	No	No
INTEGER*2	Yes	Yes	Yes	Yes
INTEGER*4	Yes	Yes	Yes	Yes
INTEGER*8	Yes	No	No	No
REAL*4	Yes	Yes	Yes	Yes
REAL*8	Yes	Yes	Yes	Yes
REAL*10	Yes	No	No	No
LOGICAL*1	Yes	Yes	Yes	No
LOGICAL*2	No	No	No	Yes
LOGICAL*4	No	Yes	Yes	Yes
COMPLEX*8	Yes	Yes	Yes	Yes
COMPLEX*16	Yes	No	Yes	Yes
COMPLEX*20	Yes	No	No	No
DECIMAL	No	No	No	Yes (library)
CHARACTER*n				
Maximum n	65,535	255	16,383	127
// operator	Yes	Yes	Yes	No
NAMES (length)				
Global	1-40	1-8	1-8	1-6
Local	1-40	1-31	1-8	1-1,320 (unique in 6)
Special symbols	$ _	$	$	None

The DRI, IBM, and Lahey compilers implement the full FORTRAN 77 language; each product incorporates its own additional features. The Microsoft compiler implements subset FORTRAN, but includes many of the features of the full language.

WEIGHING DIFFERENCES

Aside from conformance to the ANSI standard, other considerations in the choice of a FORTRAN compiler include the quality of the documentation, both for the language and for the compiler and associated support programs; the operating environment required for compiling and executing programs; its ease of use in terms of editing, compiling, loading, and testing programs; the facilities available for using procedures written in another language and for accessing the operating system services; and the execution size and speed of the code that is generated.

Results for the benchmark tests are shown in table 2. The same source code is used for each compiler. All testing was done on an IBM PC with 256KB of RAM, two DSDD floppy-disk drives, DOS 3.0, and an 8087 floating-point processor chip installed. Except for the IBM and Lahey compilers, which require an 8087, the floating-point tests were run with and without the 8087.

The benchmarks consist of a set of five FORTRAN programs that range in size from one line to 158 lines. All times were taken using the system clock. The call to **etime** (listing 1) reads the system clock, subtracts the starting time from the current time, and calculates the average time per iteration given the start time and the iteration count. This assembly language interface was used for the DRI and Microsoft systems; IBM and Lahey supply intrinsic functions to read and set the time and date (a minor aid in interfacing with the system, but better than nothing).

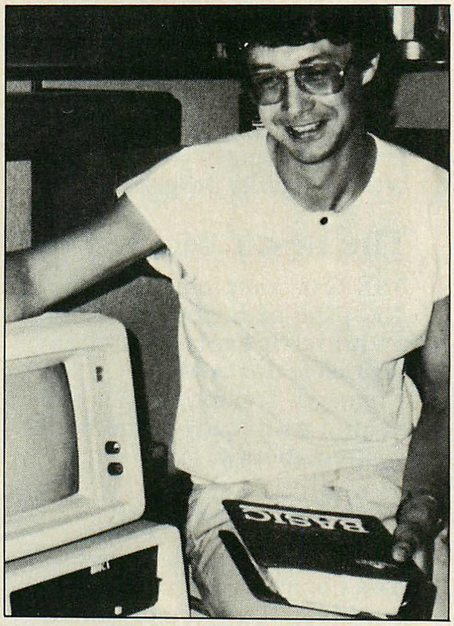
The first benchmark, MINIMUM.F77 (listing 2), is simply an END statement. While this is not a useful program, it does provide a base for evaluating the other benchmarks. The times and sizes of the compilations for this program represent the overhead required to run the compilers and linkers with a minimum executable program size. The times, of course, would be considerably less when running with a hard-disk or RAM-disk system.

TRIG.F77 (listing 3) calculates the function $sin^2x + cos^2x$ in the range $-90 <= x <= 90$ degrees. The calculations are done in single and double precision and the maximum error over

"I Program In BetterBASIC And I Recommend It."

"My company uses BetterBASIC to control the automation system on board ships. It gives us the excellent feature of flexibility and quick change of process dependant logic. We save the logic in E²prom (electrically erasable programmable ROM) so online changes can be made instantly. I use BetterBASIC and I recommend it."

Ken Moller · Norcontrol · Horten, Norway



640K Now you can use the full memory of your PC to develop large programs.
STRUCTURED Create well organized programs using procedures and functions that are easily identified and understood and completely reusable in future programs.
MODULAR Use procedures and functions grouped together to form "library modules" which are then available to you or anyone else for future use.
EXTENSIBLE Create your own BetterBASIC modules which contain BetterBASIC extensions. This feature coupled with the easy-to-use Assembly Language support, makes this an ideal OEM language.
INTERACTIVE BetterBASIC acts like an interpreter because it responds to the users' commands in an immediate mode. However each statement is actually compiled as it is entered.
COMPILED Each line of the program is compiled as it is entered into the computer's memory rather than interpreted at runtime.
RUNTIME SYSTEM The optional Runtime System generates stand alone EXE. files allowing for the distribution of products written in BetterBASIC with no royalties.
SUPPORTS Windows, Graphics, DOS and BIOS ROM calls, Chaining, Overlays, Local and Global Variables, Recursion ... and more.
BetterBASIC Runs on IBM PC, XT, AT and all IBM-compatibles. Ask your local dealer for BetterBASIC or call 1-800-225-5800 in Canada call 416-469-5244. Also available for the Tandy 1000, 1200, AND 2000 AT Tandy/Radio Shack stores.

Summit Software Technology, Inc.™
P.O. Box 99, Babson Park
Wellesley, MA 02157

PRICES:

BetterBASIC	$199
8087/80287 Math Module	$99
BTrieve™ Interface	$99
Runtime System	$250
Sample Disk with Tutorial	$10

Better BASIC.

Because It's The Best.

CIRCLE NO. 179 ON READER SERVICE CARD

MasterCard, Visa, Checks, Money Order, C.O.D. accepted and P.O. on approval.

(If you're using BetterBASIC and would like to be featured in one of our ads, please write to the Director of Advertising at Summit.)

the range is printed after the timing of the iterations. The errors in the single-precision tests were on the order of 10^{-8} except Microsoft, which was 10^{-7}. The errors in the double-precision tests were around 10^{-16}.

Listing 4, the Eratosthenes Sieve program, calculates the prime numbers in the range 1 to 16,381, then prints the number of primes and the largest prime in the range. It is primarily a test of the loop and conditional code generated by the compilers.

The pentathlon benchmark (listing 5) is a set of five programs converted to FORTRAN from the original set used by William J. Hunt in "C and the PC: Part 1" (*PC Tech Journal*, November/December 1983, p. 110). The floating-point test initializes two real arrays with implied conversion from integer to real, then calculates the inner product of the arrays. The function-call test calls a dummy integer function 20,000 times. The copy test creates a character array

The addition of features is not a problem. The problem is that each supplier may implement a feature differently.

of 127 items and copies it to a second character array 200 times. Because FORTRAN offers no string handling, this test provides timing information on how well the compilers manage character variables that could be used to write a set of string procedures. The character-count test calculates the distribution of characters in a character variable, similar in function to the copy test.

Pentathlon's file-copy test uses unformatted I/O to copy a 15,000-byte file. Only the DRI compiler produced a file exactly 15,000 bytes long. Surprisingly, the files produced by the IBM, Lahey, and Microsoft compilers were 2 to 5 times larger than the expected 15,000 bytes. The extra information in the files seems to be record-length data to support the FORTRAN BACKSPACE statement for variable-length record files. If the files are opened with the RECL= option to specify a fixed length, the extra information is not necessary.

FILECOPY.F77 (listing 6) creates direct access files of a given record length and copies the first file to the second. This test gives the actual data transfer time because the output file for the

TABLE 2: *Benchmark Results*

BENCHMARK	DRI Small	DRI Large	IBM Large	LAHEY Large	MS Large
MINIMUM (1 line)					
Compile time	0:24	0:25	0:19	0:18	0:34
Compiled size	11	12	24	90	35
Link time	1:52	2:29	0:30	1:06	1:52
Linked size	33,920	60,544	10,418	22,546	31,374
Seconds	4	6	4	5	4
TRIGTEST (49 lines)					
Compile time	0:37	0:37	0:44	0:24	0:53
Compiled size	802	921	800	978	938
Without 8087					
Link time	3:15	4:09	n/a	n/a	2:48
Linked size	75,643	95,360	n/a	n/a	44,204
Seconds					
Real*4	80.14	80.19	n/a	n/a	2.55
Real*8	80.89	80.96	n/a	n/a	7.80
With 8087					
Link time	3:06	3:46	1:10	1:26	2:07
Linked size	60,032	79,872	32,802	32,326	37,292
Seconds					
Real*4	0.66	0.70	0.27	0.24	0.53
Real*8	0.66	0.71	0.29	0.26	0.56
ERATOSTHENES SIEVE (44 lines)					
Compile time	0:39	0:35	0:39	0:23	0:47
Compiled size	439	502	501	765	507
Link time	3:00	3:24	1:15	1:31	2:10
Linked size	88,576	91,648	63,746	64,500	67,218
Seconds	1.48	1.46	1.53	2.34	2.61
PENTATHLON (158 lines)					
Compile time	1:05	1:42	1:45	0:39	1:15
Compiled size	1,632	1,890	1,683	1,654	1,308

copy has been allocated by the create phase. Only the copy time is measured. Again, the timings indicate which systems write on each call and which write only when a sector buffer is full.

FORTRAN FROM DRI

The DRI compiler implements the full FORTRAN 77 language. (Version 4.0 is reviewed here; DRI has released an improved version 4.1, but it was not available for review.)

The ANSI standard specifies that "an integer, real, or logical datum has one numeric storage unit in a storage sequence." LOGICAL*1 may cause unexpected results in equivalence statements between logical and nonlogical data types. The default for integer storage is two bytes, which again does not conform to the standard, but it can be set to four bytes with a compiler option. Double precision is the same as REAL*8.

This compiler generates 8087 coprocessor code directly; it does not support DOS 2.0 paths. This can be annoying on a hard-disk system and more so on a floppy-disk system. The use of batch files for compiling and linking helps. (Version 4.1 offers complete support for DOS path names.)

Symbolic names can be 1 to 40 characters long and the use of lowercase letters, dollar sign, and underscore is allowed. Keywords, however, must be uppercase; this includes quoted keywords in I/O statements, such as STATUS = 'NEW'. The best solution is to keep the Caps Lock key on and type everything in uppercase. A compile switch will convert nonquoted symbols to uppercase, but this does not affect the operators (.EQ., .NOT., etc).

The compiler exhibits two internal errors. The first error seems to be associated with the use of function calls as output values in a WRITE statement. In the second case, the compiler generates an error in the construction

```
SUBROUTINE X (A, SIZE)
INTEGER*2      A (SIZE), SIZE
```

The message says that SIZE is multiply defined; it is not. Reversing the order of SIZE and A(SIZE) in the declaration statement eliminates the problem.

BENCHMARK	DRI Small	DRI Large	IBM Large	LAHEY Large	MS Large
PENTATHLON (cont.)					
Without 8087					
Link time	3:20	3:28	n/a	n/a	2:12
Linked size	77,184	96,384	n/a	n/a	41,028
With 8087					
Link time	2:45	3:04	1:23	1:39	1:55
Linked size	61,568	80,256	37,890	40,484	36,808
Seconds					
Floating point					
Without 8087	11.78	11.78	n/a	n/a	0.83
With 8087	0.15	0.15	0.12	0.13	0.21
Function calls	1.92	2.55	3.63	9.07	1.88
String copy	6.12	7.86	2.84	8.57	1.85
Character count	1,210	1,215	16.02	8.55	10.59
Copy 15KB file	232	462	324	61	181
File size	15,000	15,000	75,000	30,001	45,002
FILE COPY (87 lines)					
Compile time	1:06	1:09	1:15	0:29	1:06
Compiled size	1,466	1,678	1,493	2,132	1,269
Link time	2:54	3:15	1:22	1:37	2:10
Linked size	74,624	93,568	36,562	39,874	38,496
Seconds					
64 bytes	51.39	62.97	40.37	65.20	316.24
128 bytes	51.29	50.19	40.76	64.92	150.07
256 bytes	51.39	50.10	40.37	28.18	71.69
512 bytes	40.99	40.79	40.53	28.13	40.79
1,024 bytes	14.95	15.24	14.89	21.81	39.40

Lahey's F77L consistently showed the best compile time through the benchmark tests. IBM's compiler was generally faster than the others in execution speed.

The documentation for the DRI compiler is concise and clear, with several examples. The discussion of each statement type starts with a new page, which simplifies searches.

The general level of error detection is good. Diagnostics are classed as errors or warnings. One extremely useful warning is the flagging of the first instance of a name that has not been defined. Unfortunately, nothing in the documentation indicates which messages are errors and which are warnings; only the compiler makes this clear (warning messages are displayed if the -w compile switch is enabled). This distinction would be a useful addition to later versions of the documentation.

No explanations are given for error messages. The preface to the error message list says that they are self-explanatory, but not all of them are. It would be very helpful, for example, if the conditions for an undefined symbol reference were explained. Runtime errors are displayed with English text if the runtime error file is located on the current disk.

DRI FORTRAN is the only compiler tested that supports both the large (unlimited code and data) and small (64KB code, 64KB data) memory models. The others support the large model only. The two DRI models are incompatible, so one or the other should be used exclusively. Two runtime libraries are supplied, and the correct one must be selected at link time.

IBM FORTRAN

IBM's entry implements the full FORTRAN 77 language. IBM FORTRAN was written by Ryan-McFarland Corporation and is currently being sold by that company's distributors as well as by IBM's. The documentation for this compiler is by far the best for the products reviewed here and better than most of the documentation written for mini or mainframe FORTRAN compilers. The debugger included with IBM FORTRAN is the most comprehensive of the three compilers that have a debugger.

This compiler requires an 8087 coprocessor, DOS 2.1 or later, and 192KB of memory. A minimum of 64KB of

memory is required for execution, plus 90KB if the debugger is used.

IBM FORTRAN is a conservative implementation of the full language. It has relatively few extensions, and most of them are patterned after IBM mainframe FORTRAN compilers. One nice extension is the inclusion of intrinsic functions to access and change the system date and time. An extension that would be welcome is the relaxation of the ANSI standard on mixing character and noncharacter data types in common blocks and equivalence statements.

One internal error surfaced during the development of the benchmarks. The compiler indicated an error on the following code fragment, which uses the new substring notation to assign the value of *i* to the *i*th element of a 127-element character array:

```
do 1 i = 1, 127
      s (i: i) = char (i)
1 continue
```

This change solved the problem:

```
do 1 i = 1, 127
      j = i
      s (i: i) = char (j)
1 continue
```

Runtime errors are reported by error code only, requiring ready access to the documentation. Compile time errors, on the other hand, are quite clear; they are inserted in the listing close to the offending source statement.

The documentation is logically formatted and contains many examples; nonstandard features are noted. The information on the intrinsic functions includes the input and output ranges, the algorithm, possible runtime errors, any other support functions called, and the amount of memory required. Clear installation instructions are provided for systems with dual floppy disks, 1.2MB drives, and hard-disk drives.

Compiler and linker usage is very straightforward; the compile options are specified on the directive line and only one runtime library exists. Linkage of the debugger is automatic when the main program is compiled with the debug option. However, without a hard disk, loading a program with the debugger requires many swaps of the runtime and debug library diskettes. The few options that are provided are adequate.

The compiler comes with an easy-to-use source level debug package that nearly eliminates the need to search the listings to look at machine code. The user has only a few commands to remember and the sole drawback is that the program source code is not displayed. The HELP command reduces the need to have the debugger manual open during a session.

The main program of a group to be debugged, as well as selected subprograms, must be compiled with the debug option. The program is then relinked and executed. When the program has started, the user enters debug commands to monitor its execution. Both trace and break commands are provided. Breaks can be set to suspend execution when a variable is changed or when a specific condition exists. As soon as a break is reached, the current location is displayed (by name and line number) and the current traceback information may be displayed. Variables can be examined and changed by name; it is not necessary for the user to know the variable's address.

IBM FORTRAN generates code for the large memory model. Compiled code size was average for the benchmark tests, and IBM's execution was faster than the other compilers overall.

LAHEY'S F77L

The F77L compiler, from Lahey Computer Systems, Inc., implements the full FORTRAN 77 language with several extensions (although they are not designated as such). The documentation is

application

Assembler speed in IBM Basic and Microsoft "C".
Mainframe power for your IBM PC.
Create network applications.*
Choice of file or record locking.

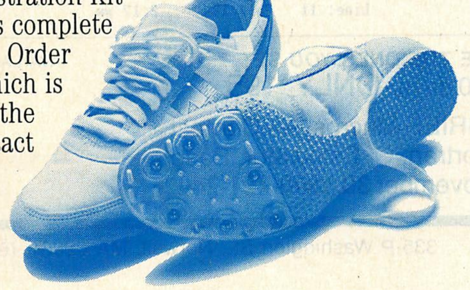
Today's programmers want more than development tools and Data Base Managers that merely crawl when they need to sprint. APPLICATION 1's rich assembler routines combined with IBM Basic or Microsoft "C", sprints during the creation and execution mode of your software...and that means up to 6 times faster and more!

Its remarkable efficiency is a result of this unique file system. Achieve unperceivable performance differences from files containing 300 to as many as 80,000 records or more. Design screens in minutes with Genscr, our full screen editor. And most important, APPLICATION 1 opens up the door to file sharing in update mode. Now.

Conserve precious disk space with variable length record and key indexes. With unlimited files, fields, records and 8087/80287 support, APPLICATION 1 has it all.

You can grow with us because we bring no limitation to your imagination and future requirements.

We've prepared a "Demonstration Kit" for Application 1. This includes complete documentation and demo disk. Order yours today for $49.95 (US) which is refundable on the purchase of the $595.00 (US) full version. Contact Logiquest at 514-397-9631.



LOGIQUEST

Logiquest Inc.
555 Dorchester West, Suite 1616
Montreal, Quebec, Canada H2Z 1B1
1-514-397-9631

Required:
PC DOS 2.1 and up
PC DOS 3.1 and up for networking
256k Ram.
IBM PC, XT, AT, and compatible
IBM Network Cards presently supported*

©APPLICATION 1 is a trademark of Logiquest Inc., Montreal. IBM, IBM Basic are products and trademarks of International Business Machines. Microsoft is a trademark of Microsoft Corporation.

CIRCLE 202 ON READER SERVICE CARD

good but suffers from excessive typographical errors. The debugger is limited to tracing the flow of execution of subprogram calls, with no provision for examining or altering data values. These limitations notwithstanding, F77L is the clear leader in compile speed. It requires an 8087 and 256KB of RAM.

Source files may be in either the standard FORTRAN format or in a free format. In the free-format mode, comments are specified with an asterisk in column one, and continuation lines are specified with a plus sign. Utility programs are supplied to do conversions between the two formats.

Source files are compiled with a single command (**F77L file**); optional compiler directives may be included on the command line. An interesting feature of this compiler is a standard directive file, which contains the default compile options. A useful extension to this concept would be for F77L to accept a list of include files in the compiler directive file to allow the inclusion of a project standard set of parameters in each program unit.

The standard DOS linker is used. The runtime executive file (F77L.OBJ) must be loaded explicitly as the first relocatable file; therefore, the name of the executable file must be specified.

A source on-line debugger (SOLD) is included with the compiler package. SOLD operates on the output of the linker without recompiling or relinking the programs. The functions supported by SOLD are limited to tracing execution through specified segments of a program and stopping execution at selected locations. Break and trace locations are the entry and exit of subprograms or at specific line numbers within a subprogram. However, SOLD offers no provision for examining and changing data items, and the compiler does not generate a source listing, which makes the search for line numbers in the source file tedious.

Programs that are compiled with F77L may be linked with procedures that have been compiled with the Lattice C compiler.

System functions are supplied to generate random numbers, to locate the position of the last nonblank character in a string, and to return a string with the trailing blanks removed. System subroutines are supplied to access the system date and time, type a message and give a subprogram traceback, get a copy of the execution command line, and execute a copy of COMMAND.COM with the command to execute being passed as an argument.

MICROSOFT FORTRAN

Microsoft's product implements the subset FORTRAN 77 language. Many full-language features also are included, the most important of which are: BLOCK DATA, array references as subscripts, expressions in the I/O list of a WRITE statement (the expression cannot start with a left parenthesis), DOUBLE PRECISION and COMPLEX numbers, and restricted CHARACTER typed functions. However, character expressions are not implemented (subfields are supported in version 3.3). Implied DO loops and the ENTRY statement are also missing.

MS-FORTRAN has a format operator ($) to suppress the carriage return at the end of an output statement, which is useful for screen input forms. All of the other compilers, however, suppress the carriage return at the end (consistent with the ANSI standard that the column-one formatting control acts before the data transfer).

Integer constants in any radix from 2 to 36 may be expressed using the notation <radix>#<digits> where <radix> is the radix and <digits> is a sequence of digits in the range [0, radix−1]. The letters *A* through *Z* are used for 10 to 35.

The MS compiler allows character and noncharacter variables to be in the

same common block and to be made equivalent; however, noncharacter variables must start on an even byte boundary. This is a problem primarily in common blocks but it can be avoided by putting the character variables at the end of the common block. Conflicts trigger an error message.

The package includes a compile time option ($DECMATH) and runtime library (DECMATH.LIB) to support decimal floating-point numbers (either 6 or 14 digits may be expressed exactly).

A relational database system with nearly 15,000 lines of FORTRAN code has been converted from the DEC PDP/11 and DEC VAX to the PC/XT using MS-FORTRAN. Only one compiler error was found: the MS product sometimes indicated an error when calling an INTEGER*4 function with an array name as an argument. Another unresolved problem involved linking FORTRAN named common blocks to assembly language programs.

The Microsoft documentation comes in two volumes and is generally quite clear. Separate indexes for the user's guide and reference manual require a lot of page flipping. The indexes seem incomplete; they offer no reference, for example, to the use of numbers with radix other than 10.

This compiler's error detection is good; however, it has problems recovering after a source code error and often generates incorrect error messages before it returns to normal.

The MS compiler requires two separate programs to compile a program, three if an assembly language source listing is required. Each section of the compiler must be executed explicitly with a command. A batch file to execute the first and second passes and passing the file names as parameters is a simple way to make this less tedious. Only the input and output files can be specified on the command line. Any compiler directives, such as two-byte integers as default, must be specified with metacommands in the source file. This is not very convenient as they must be specified in every source file instead of once in a standard batch file. MS-Pascal and MS-C subroutines and functions can be linked with MS-FORTRAN modules.

Using the default conditions, all floating-point operations are compiled as calls to the runtime floating point library. Generation of in-line 8087 code requires that the $NOFLOATCALLS metacommand be included at compile time. The requirement to include compiler directives of this type—that is, a directive that affects an entire system—would be more conveniently specified in the compiler command line.

The debugger available with the Microsoft assembler (although not included with the FORTRAN compiler) is nearly as good as the IBM FORTRAN debugger. It permits displaying the source code that is being debugged, but finding program variables is very awkward. The trace commands execute on an instruction-by-instruction basis; it would be better if they executed a source line.

Using the Microsoft compiler or any of these products for serious program development requires a hard disk or a large RAM disk because of the size of the compilers, libraries, and loaders.

The IBM compiler's documentation, ease of use, speed of execution, and debugger facilities place it first for recommendation. A good second choice is Lahey's F77L, even though its documentation needs revision. The choice between the Microsoft and DRI compilers is a decision that users will probably base on other software that is supported by the product's vendor: DRI FORTRAN can be combined with DRI C, whereas MS/FORTRAN can be combined with MS/Pascal.

FORTRAN-77 4.0: $350
Digital Research, Inc.
60 Garden Court
Box DRI
Monterey, CA 93942
800/443-4200
CIRCLE 349 ON READER SERVICE CARD

IBM PC Professional FORTRAN 1.0: $595
IBM Corporation
P. O. Box 1328
Boca Raton, FL 33432
800/447-4700
CIRCLE 350 ON READER SERVICE CARD

F77L 1.1: $477
Lahey Computer Systems, Inc.
31244 Palos Verdes Drive West
Suite 243
Rancho Palos Verdes, CA 90274
213/541-1200
CIRCLE 351 ON READER SERVICE CARD

Microsoft FORTRAN 3.2/3.3: $350
Microsoft Corporation
10700 Northup Way
Bellevue, WA 98004
800/426-9400
CIRCLE 352 ON READER SERVICE CARD

Alan Howard is the principal scientist with Contel Information Systems in Bethesda, Maryland. He has a master of mathematics from San Jose State University and earned his bachelor's degree at Stanford University.

Which Would You Prefer?

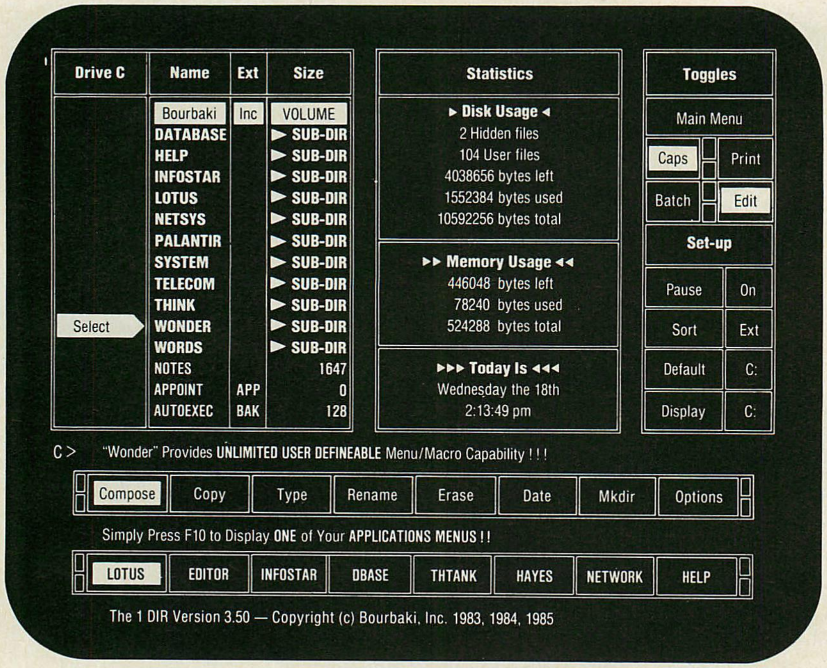
A>_

This

or

This

Customize Your System
with the Ultimate
"Productility"



the 1 dir

"Wonder"

EASY to USE "VISUAL SHELL" for DOS

"WONDER" integrates a powerful File Management System with extensive user definable Menu/Macro capability. Create menus and program single keystroke commands to run any or all of your applications. Create your own customized HELP system for yourself or other users. In addition, version 3.5 provides full color support, menu password protection, CUSTOM multiple file operations, CUSTOM prompted commands, plus much more.

If you are a member of the growing population of new PC users, whose interest lies primarily in taking advantage of the computer as information management tool, this program was designed for you. If you have a hard disk, you shouldn't be without it! ***1 dir*** is licensed by more hard disk manufacturers than any other visual shell.

MICRO D Corporate Training Specialist — Mr. Don Kenny

"What impresses me most about ***1 dir*** is how much you have intelligently added to it over the last few months. It is very rare to find a company such as yours with a practical knowledge of how computers can be tamed . . . ***1 dir*** is still the **BEST** operating environment I've ever seen on a computer."

"WONDER" is also a *perfect* Training Tool that grows *with* the *user!*
A real *problem solver* for MIS Managers and Systems Integrators.

Works with the IBM PC/XT/AT family. (DOS 2.00 — or higher), plus compatibles.

Call, write or see your local dealer for more information!
Evaluation copies available to dealers and corporate clients.

by Bourbaki Inc.™

$95.00

Lotus is a trademark of the Lotus Development Corporation.

CIRCLE NO. 235 ON READER SERVICE CARD

P.O. Box 2867, Boise, ID 83701, (208) 342-5849

LISTING 1: ETIME.F77

```
      subroutine etime (e,     s,      display)
      integer*2          e (4), s (4)
      integer*2                        display

      integer*2   i, now (4)
      real*4      time bar, average
      character   punct (4)

      data        punct  / ' ', ':', ':', '.'/

      call time (now)
      call subtime (e, s, now)
      if (display .ge. 0) then
         write (*, 98) (punct (i), e (i), i = 1, 4)
         if (display .gt. 0) then
            average = time bar (e, display)
            write (*, 99) display, average
            end if
         end if
      return

98    format (' elspsed time', 4(a1, i2))
99    format (' average time for ', i3, ' iterations',
     1        f9.3, ' seconds')
      end
      subroutine subtime (delta, start, stop)
      integer*2   delta (4), start (4), stop (4)

      integer*2   i, borrow
      integer*2   adjust (4)

      data        adjust / 24, 60, 60, 100/

      borrow = 0
      do 1 i = 4, 1, -1
         delta (i) = stop (i) - start (i) - borrow
         if (delta (i) .lt. 0) then
            delta (i) = delta (i) + adjust (i)
            borrow = 1
         else
            borrow = 0
            end if
1     continue
      return
      end
      real*4        function time bar (time, n)
      integer*2     time (4)
      integer*2     n

      integer*4     hseconds
      integer*4     convert (4)
      integer*2     i
      data          convert / 60, 60, 100, 1/

      hseconds = 0
      do 1 i = 1, 4
         hseconds
     1      = convert (i)*(hseconds + time (i))
1        continue

      time bar = float (hseconds) / float (n) / 100.0
      return
      end
```

LISTING 2: MINIMUM.F77

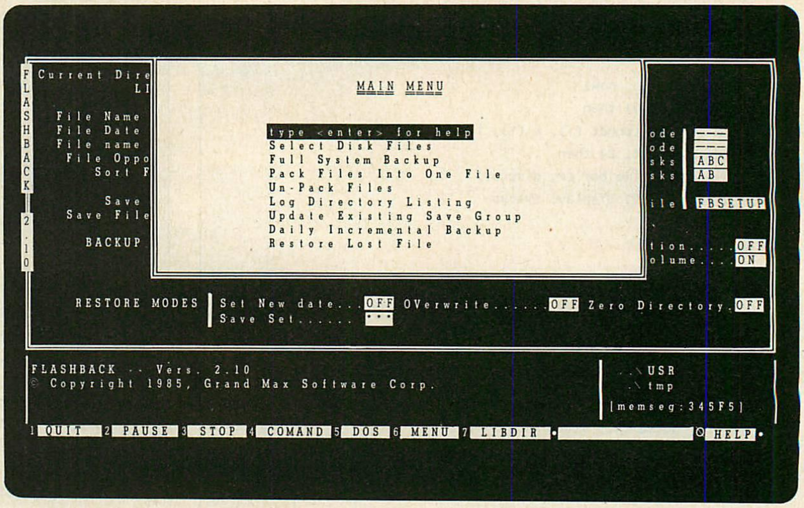
```
      END
```

LISTING 3: TRIG.F77

```
C     trig test
C                      2         2
C     calculate sin  x + cos  x for -90 <= x <= 90
C
C     the maximum error is saved and printed after 'iters'
C     calculations in real*4 and real*8

      program     trig test

      real*4      x,   sx,   cx,   error,   radians
      real*8      d x, d sx, d cx, d error, d radians
      integer*2   iters
      integer*2   max iters
      integer*2   i
      integer*2   t1 (4), t2 (4)

      radians (i)   = float (i)*0.01745 32925
      d radians (i) = float (i)*0.01745 32925 19943 D 0

      write (*, *)' iters: '
      read  (*, *) max iters
      call time (t1)
      do 200 iters = 1, max iters
         error = 0.0
         do 100 i = -90, 90
            x   = radians (i)
            sx  = sin (x)
            cx  = cos (x)
            error = max (error, abs (1.0 - sx**2 - cx**2))
100         continue
200      continue
      call etime (t2, t1, max iters)
      write (*, *)'real*4 error', error
      write (*, *)' '

      call time (t1)
      do 400 iters = 1, max iters
         d error = 0.0
         do 300 i = -90, 90
            d x  = d radians (i)
            d sx = sin (d x)
            d cx = cos (d x)
```

FILE MANAGEMENT is *Fast Backup* and much, much more...

FILEMASTER supports over 60 different file management commands, plus an easy to use menu system for executing routine command sequences.



FILEMASTER

THE DEFINITIVE BACKUP AND FILE MANAGEMENT SYSTEM FOR HARD DISK USERS.

FILEMASTER is a complete set of tools for managing the complexities of the DOS file system.

FILEMASTER'S fast and versatile backup program is only part of the story. With larger fixed disk media becoming commonplace, full featured FILE SYSTEM MANAGEMENT is becoming a requirement on personal business computers.

The more files you create, the more you need to keep them organized. With over 60 different file management commands, FILEMASTER helps you get control of your file system, and keep it organized, no matter how busy you get.

A versatile menu system lets you set up routine maintenance procedures that anyone can perform, even those who are unfamiliar with your computer.

You may only use FILEMASTER a few minutes each day; but those few minutes will pay dividends all day long.

FILEMASTER is now available for the IBM PC-AT, and DOS version 3.0 & 3.1, for only:

$49.95

plus $5 for shipping & handling. The package includes a 180 page user manual, plus a set of on-line tutorials, and a support line for technical assistance.

Hardware Requirements: IBM-PC XT, AT, Compaq-Plus, or compatible equipment; DOS versions 2.0-3.1; minimum 128k, some programs require 256K.

Shipping prices for delivery in U.S.A. only.

FILEMASTER by Grand Max should not be confused with fileMASTER, a separate product marketed by J. Schuller & Assoc.

FILEMASTER is available only from Grand Max Software Corporation. To order with VISA or Mastercard, dial:

1-800-624-3285

In Oregon: (503) 635-8295

Grand Max™
SOFTWARE CORPORATION

68 LEONARD STREET • LAKE OSWEGO, OR 97034

```
      d error = max (d error, abs (1.0 - d sx**2 - d cx**2))
300       continue
400    continue

      call etime (t2, t1, max iters)
      write (*, *) 'real*8 error', d error
      end
```

LISTING 4: SIEVE.F77

```
C     prime number sieve program

      integer*2   i, niter, count, prime
      integer*2   t1 (4), t2 (4)

      write  (*, '  ('' no. iterations: '' ) . ')
      read   (*, *) niter

      call time (t1)
      do 30 i = 1, niter
         call sieve (count, prime)
30    continue

      call etime (t2, t1, niter)
      write  (*, 40) count, prime
40    format ('0 done', I6, ' primes, largest is ', I6)
      end

      subroutine sieve (count, largest)
      integer*2         count, largest

      integer     size
      parameter   (size = 8191)

      integer*2   i, prime, k
      logical     flags (size)

      count = 0
      do 10 i = 1, size
         flags (i) = .true.
10       continue

      do 30 i = 1, size
         if (flags (i)) then
            prime = i + i + 1
            do 20 k = i + prime, size, prime
               flags (k) = .false.
20             continue
            count = count + 1
            end if
30       continue
      largest = prime
      return
      end
```

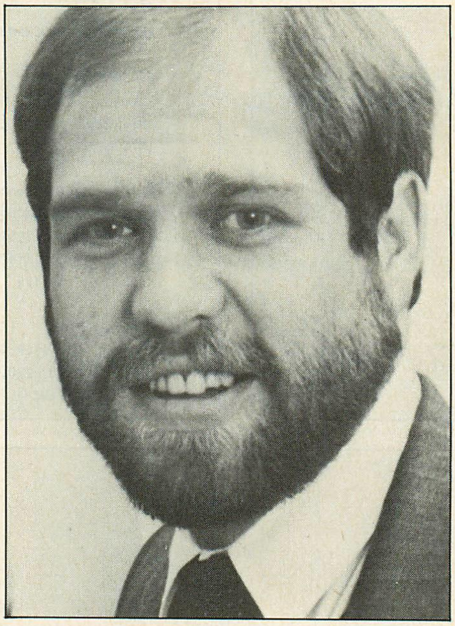
LISTING 5: PENT.F77

```
C     SOURCE FILE:   PENT.F77
C     EDIT DATE:     18MAR85
C     AUTHOR:        A. J. HOWARD
C
C     PENTATHLON PROGRAM, FORTRAN 77 VERSION

      PROGRAM PENT

      INTEGER*2   I, ITER, BENCH
      INTEGER*2   E (4), S (4)

1     WRITE  (*, *)' Benchmark: '
      READ   (*, *)BENCH
      IF (BENCH .LE. 0)STOP
      WRITE  (*, *)' Iterations: '
      READ   (*, *)ITER

      CALL TIME (S)
      DO 30 I = 1, ITER
         GO TO (100, 200, 300, 400, 500), BENCH
100         CALL BENCH1
```

"I Program In BetterBASIC And I Recommend It."

"I use BetterBASIC over C and PASCAL because it's more versatile and convenient. My department acts as an in-house consultant at Fannie Mae. We evaluate software for the entire organization. When the need arises for a programming language, I recommend BetterBASIC."

Lee Beckley · Information Center Analyst · Fannie Mae · Washington, D.C.



640K Now you can use the full memory of your PC to develop large programs.
STRUCTURED Create well organized programs using procedures and functions that are easily identified and understood and completely reusable in future programs.
MODULAR Use procedures and functions grouped together to form "library modules" which are then available to you or anyone else for future use.
EXTENSIBLE Create your own BetterBASIC modules which contain BetterBASIC extensions. This feature coupled with the easy-to-use Assembly Language support, makes this an ideal OEM language.
INTERACTIVE BetterBASIC acts like an interpreter because it responds to the users' commands in an immediate mode. However each statement is actually compiled as it is entered.
COMPILED Each line of the program is compiled as it is entered into the computer's memory rather than interpreted at runtime.
RUNTIME SYSTEM The optional Runtime System generates stand alone EXE. files allowing for the distribution of products written in BetterBASIC with no royalties.
SUPPORTS Windows, Graphics, DOS and BIOS ROM calls, Chaining, Overlays, Local and Global Variables, Recursion ... and more.
BetterBASIC Runs on IBM PC, XT, AT and all IBM-compatibles. Ask your local dealer for BetterBASIC or call 1-800-225-5800 in Canada call 416-469-5244. Also available for the Tandy 1000, 1200, AND 2000 AT Tandy/Radio Shack stores.

Summit Software Technology, Inc.™
P.O. Box 99, Babson Park
Wellesley, MA 02157

PRICES:

BetterBASIC	$199
8087/80287 Math Module	$99
BTrieve™ Interface	$99
Runtime System	$250
Sample Disk with Tutorial	$10

Better BASIC.

Because It's The Best.

CIRCLE NO. 195 ON READER SERVICE CARD

MasterCard, Visa, Checks, Money Order, C.O.D. accepted and P.O. on approval.

(If you're using BetterBASIC and would like to be featured in one of our ads, please write to the Director of Advertising at Summit.)

```
      GO TO 30
200   CALL BENCH2
      GO TO 30
300   CALL BENCH3
      GO TO 30
400   CALL BENCH4
      GO TO 30
500   CALL BENCH5
30    CONTINUE
      CALL ETIME (E, S, ITER)
      WRITE (*, *)' DONE- '
      GO TO 1
      END
C     FLOATING POINT ARITHMETIC BENCHMARK
      SUBROUTINE BENCH1
      INTEGER*2  I, J
      REAL       X (100), Y (100), Z
      DO 10 I = 1, 100
        X (I) = I + 1
        Y (I) = 3*I
10      CONTINUE
      Z = 0.0
      DO 20 J = 1, 10
        DO 20 I = 1, 100
          Z = Z + X (I)*Y (I)
20        CONTINUE
      RETURN
      END
C     FUNCTION CALLING BENCHMARK
      SUBROUTINE BENCH2
      INTEGER*2  I, RESULT
      INTEGER*2  DUMMY
      DO 10 I = 1, 20000
        RESULT = DUMMY (I)
10      CONTINUE
      RETURN
      END
      INTEGER*2 FUNCTION DUMMY (I)
      INTEGER*2            I
      DUMMY = I + 1
      RETURN
      END
C     STRING COPY BENCHMARK
      SUBROUTINE BENCH3
      INTEGER*2      I, J
      CHARACTER*127  S, S2
      DO 10 I = 1, 127
        S (I: I) = 'A'
10      CONTINUE
      DO 20 I = 1, 100
        DO 20 J = 1, 127
          S2 (J: J) = S (J: J)
20        CONTINUE
      RETURN
      END
C     CHARACTER COUNT BENCHMARK
      SUBROUTINE BENCH4
      INTEGER*2      I, J, COUNT (64)
      CHARACTER*127  S
```

```
      DO 10 I = 1, 127
         J = I
         S (I: I) = CHAR (J)
10       CONTINUE

      DO 20 J = 1, 200
         CALL COUNTC (S, COUNT)
20       CONTINUE

      RETURN
      END


      SUBROUTINE COUNTC (S, COUNT)
      CHARACTER*127       S
      INTEGER*2              COUNT (64)

      INTEGER*2    I, C

      DO 10 I = 1, 127
         C = MOD (ICHAR (S (I: I)), 63) + 1
         COUNT (C) = COUNT (C) + 1
10       CONTINUE

      RETURN
      END
C     FILE COPY CHARACTER BY CHARACTER
      SUBROUTINE BENCH5

      INTEGER*2    IN, OUT
      INTEGER*2    N
      CHARACTER    C

      IN = 1
      OUT = 2
      OPEN (IN,  FILE   = 'A:TEST.IN',
     1           STATUS = 'OLD',
     1           FORM   = 'UNFORMATTED')
      OPEN (OUT, FILE   = 'A:TEST.OUT',
     1           STATUS = 'NEW',
     1           FORM   = 'UNFORMATTED')

      N = 0
10    READ (IN, END = 20, ERR = 110) C
      N = N + 1
      WRITE (OUT, ERR = 120) C
      GO TO 10

20    WRITE  (*, *) N, ' CHARACTERS'
      CLOSE (IN)
      CLOSE (OUT)
      RETURN

110   WRITE  (*, *)'*** Read Error ***'
      STOP

120   WRITE  (*, *)'*** Write Error ***'
      STOP
      END
```

LISTING 6: FILECOPY.F77

```
      program filecopy

      integer*4    n
      integer*4    buffer (256)
      integer*4    words, iter, bytes

      write  (*, 1)
1     format (' iterations, record length: ' )
      read   (*, *) iter, bytes

      do 10 n = 1, 256
10       buffer (n) = n

      words = bytes/4
      call make file ('A:FILE.IN',  buffer, words)
      call make file ('A:FILE.OUT', buffer, words)
```

```
      call doit (buffer, words, iter)
      stop
      end
      subroutine doit (buffer, size, iter)
      integer*4        size, buffer (size), iter

      integer*4   i, in, out
      integer*4   n
      integer*4   bytes, records
      integer*4   recnr, ioerr
      integer*2   e (4), s (4)

      bytes = size*4
      records = 32768 / bytes
      in  = 1
      out = 2

      call time (s)
      do 50 i = 1, iter
         open (in, file = 'A:FILE.IN',
     1          access = 'DIRECT', recl = bytes,
     1          status = 'OLD', form = 'UNFORMATTED')
         open (out, file = 'A:FILE.OUT',
     1          access = 'DIRECT', recl = bytes,
     1          status = 'OLD', form = 'UNFORMATTED')

         n = 0
         do 30 recnr = 1, records
            read (in, rec = recnr) buffer
            write (out, rec = recnr) buffer
            n = n + bytes
30       continue

40       close (in)
         close (out)
50       continue
      write (*, 60) recnr - 1, n
60    format (' done - ', I6, ' records ', I6, ' bytes')
      call etime (e, s, iter)
      return

100   write (*, 101) ioerr, recnr, n
101   format (' *** read error ', 3I6, ' ***')
      stop

200   write (*, 201) n, i, bytes
201   format ('*** write error record', I6, ' of', I6, I6, ' bytes')
      stop
      end
      subroutine make file (file, buffer, size)
      character*(*)        file
      integer*4            size, buffer (size)

      integer*4   records, out
      integer*4   n, io status
      integer*4   bytes

      bytes = size*4
      out = 2
      open (out, file = file, status = 'NEW',
     1     access = 'DIRECT', recl = bytes,
     1     form = 'UNFORMATTED')

      records = 32768 / bytes
      write (*, 15)records, bytes
15    format (' ', I8, ' records of ', I4, ' bytes')
      do 20 n = 1, records
20       write (out, rec = n) buffer
      close (out)
      return
      end
```

COMMON SENSE IN DATABASE MANAGEMENT
NUMBER TWO IN A SERIES.

SOMETIMES YOU CAN AFFORD TO FINISH IN SECOND PLACE. SOMETIMES YOU CAN'T.

When you're comparing database software, there are all kinds of things to think about. Ease of use. Power. Application development time.

Well, we think execution speed should be way up on your list, too. Because the last thing you need is a program that runs in slow motion.

That's why we've designed new R:base™ 5000 to outrun every other DBMS package available. For example, just look how it compares to dBASE III:

EXECUTION-TIME BENCHMARKS:
R:base 5000 vs. dBASE III

Using an IBM PC XT, we ran a series of performance benchmarks comparing the speed of R:base 5000 to dBASE III. The functions performed were identical. The execution times were a different story.

Function	R:base 5000	dBASE III
SORT 1,000 rows of 8 columns each on a dollar field by amount.	26 seconds	65 seconds
SEARCH 1,000 rows for unique number in non-keyed field.	7 seconds	14 seconds
REPORT with one lookup to a second table stored to a file.	6 minutes, 49 seconds	11 minutes, 8 seconds
PRINT a report of sorted data with subtotals.	6 minutes, 39 seconds	10 minutes, 30 seconds
TOTAL (All functions performed in sequence.)	14 minutes, 1 second	22 minutes, 57 seconds

R:BASE 5000. NOBODY OUTRUNS IT.

As you can see, for almost every important individual function, R:base 5000 is 60% to 150% faster than dBASE III. But that's only part of the story. If you're performing a typical operation that calls for searching, sorting, reporting and printing, then R:base 5000 puts you more than 8 minutes ahead of dBASE III.

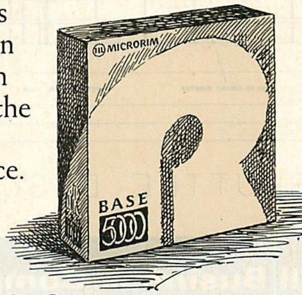
And we're just talking about the time you save on one report. As the years go by, you'll save that much a thousand times over. Since we also help you develop applications up to ten times faster, you'll be even farther ahead.

SEE FOR YOURSELF: 1-800-547-4000.

If you believe in common sense as much as we do, you won't buy something this important without trying it out first. And that's just what we'd like you to do. For only $9.95 (plus shipping) we'll send you a mini-version of the product that lets you build real-life applications.

Just call 1-800-547-4000 and ask for Dept. 818 . From Oregon, or outside the U.S., call 1-503-684-3000, Dept. 818 . We'll send your copy right out. If you'd like to see R:base 5000 today, head straight for a leading software store or computer dealer. If you own R:base 4000, ask your dealer for a trade-up kit.

We'll show you how to earn all the rewards of coming in first. Which sure beats the prize for second place.



R:BASE 5000
FROM MICRORIM
IT ALL COMES DOWN TO COMMON SENSE.

RUNTIME VERSION NOW AVAILABLE.

dBASE III is a trademark of Ashton-Tate.
IBM is a trademark of International Business Machines, Inc.

AW . . .
WHAT THE HECK!

ProDesign II
The Easy to Use CAD System!

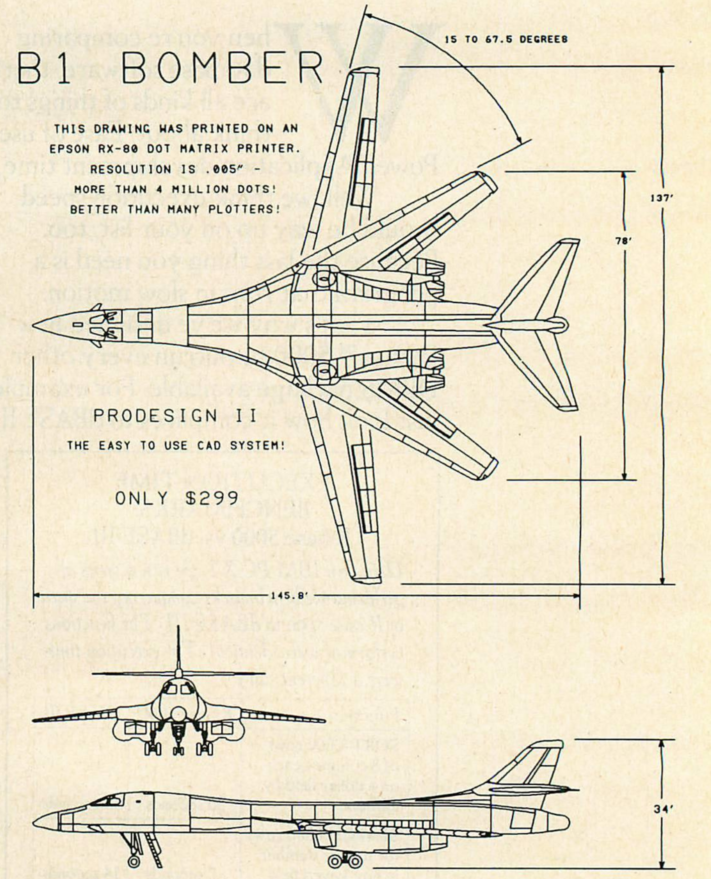
ProDesign II is one of the most advanced CAD packages available for microcomputers. We think it's absolutely the easiest to use. With competitive CAD systems priced at $1500 to $2500, we were posed with the problem of setting our price.

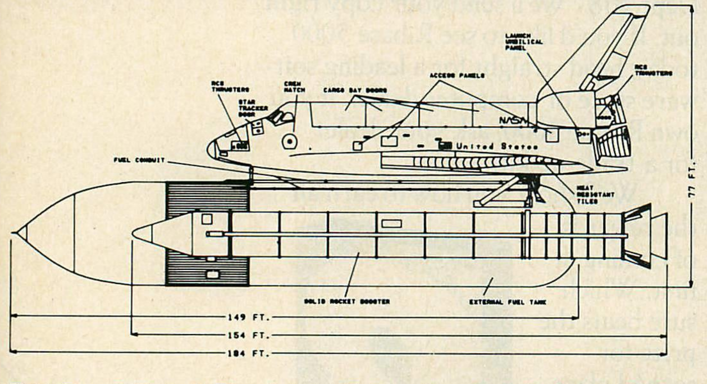
ProDesign II works a wide variety of digitizers and mouse devices. It works with nearly any plotter or printer available for the IBM PC. ProDesign II can produce plotter quality drawings on ordinary dot matrix printers - a feature found exclusively on ProDesign II. ProDesign II utilizes a virtual screen 4 times the size of the physical screen to make it practical to produce drawings on a normal resolution IBM monitor. ProDesign II is truly an outstanding CAD package for the IBM PC and compatibles. The question we had to answer was: Even though we had a better product, should we price it higher than the other CAD systems on the market?

We did market studies and calculations. We consulted with experts. We drew charts and graphs. We used the finest spreadsheet programs money could buy. When it came right down to it, we still didn't know what to sell ProDesign II for. $2995? $2495? $1995? We even considered $995.

Then, in the great American tradition, we said, "AW...WHAT THE HECK! Let's see the other guys beat this price!" ProDesign II costs $299. At that price, you can't go wrong!

AW...WHAT THE HECK!
$299.95





THE SPACE SHUTTLE DISCOVERY

American Small Business Computers
118 South Mill
Pryor, Oklahoma 74361
918/825-4844

CIRCLE NO. 103 ON READER SERVICE CARD

Why should you get ProDesign II? Four simple reasons:

1. ProDesign II is easy to use. You won't have to spend weeks learning simple functions.
2. ProDesign II works with the hardware YOU own. ProDesign II supports most printers and plotters available for the IBM PC, as well as a wide variety of digitizers and mouse devices.
3. ProDesign II can produce plotter quality output on ordinary dot matrix printers. (The B1 Bomber above was printed on an Epson RX-80.)
4. ProDesign II is priced 70% to 80% below competitive products!

What do you need to run ProDesign II? An IBM PC or compatible with 512K RAM and graphics capability.

How do you get ProDesign II? See your local computer dealer or contact us.

ProDesign II - The Easy to Use CAD System!

STEVEN ARMBRUST and TED FORGERON

A Slimmer Pascal

These seven steps reduce the size and increase the speed of Pascal programs.

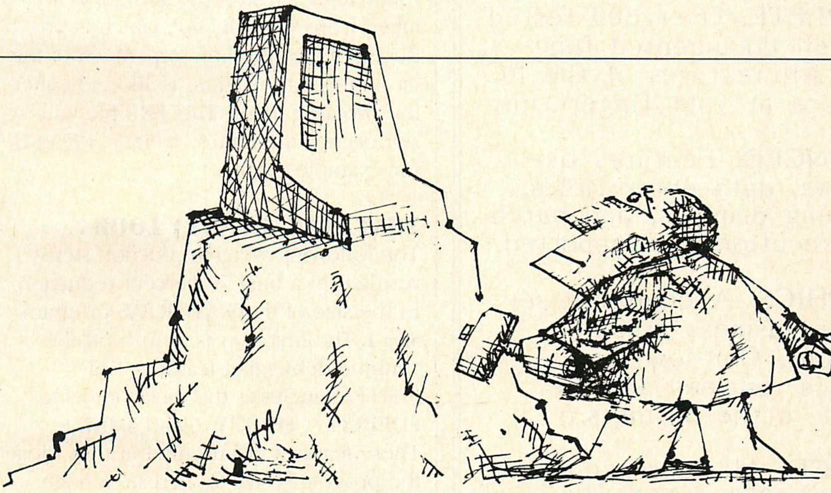


ILLUSTRATION • MACIEK ALBRECHT

Microsoft Pascal has a weight problem. Even the smallest program, a BEGIN and an END statement, for example, which yields about 20 bytes of object code after compiling, grows to over 18KB after linking to the Pascal library. If it performs any kind of task, a final in-memory program size of 80KB or 90KB is likely. This ballooning was more of a problem a few years ago when the standard PC memory was 64KB. Now 256KB, 320KB, or 512KB configurations are common. So why be concerned when a program that should occupy only 20KB actually requires over 100KB?

Three good reasons to strive for smaller programs, regardless of system configuration, are as follows. First, to risk stating the obvious, smaller usually means faster. Less code means faster loading from disk to memory, and fewer instructions result in a reduced execution time.

This is true to a point, then the classic size/performance trade-off begins to take effect. For example, with the extensive use of overlays, it is possible to create a program that requires only a tiny amount of memory. However, the performance of such a program suffers because of the time that is wasted constantly swapping modules into memory from disk. In this case, a smaller memory requirement would result in a slower execution.

The second reason is that many programs (such as print spoolers, keyboard handlers, and alarm clocks) stay resident in memory after being invoked. If each of these programs requires 100KB of memory, few persons would use them, even on fully expanded PCs. One or two such monsters could make it impossible to run any other meaningful program.

Third, the integrated environments, such as IBM's TopView, Quarterdeck's DESQ, and Microsoft's MS-Windows, are lumbering in the shadows. Soon they may find their way into a significant number of PCs. For many programs to share one integrated environment, each must use as little memory as possible.

This article describes seven steps that can greatly reduce a program's final memory requirement, making it faster as well. These steps apply to IBM Pascal 2.0 and Microsoft Pascal 3.11 to 3.2, but, with some modification, they can be applied to any Pascal program.

The program used to illustrate the techniques is a simulation of life in a simple world covered by water and inhabited by fish and sharks. (See the sidebar.) Originally called WATOR.PAS, the program has been renamed for clarity. It is listed in its fat version (FATFISH.PAS) and its slim version (SLIMFISH.PAS) on PCTECHline, as is the fat version of the utility file (FATUTIL.ASM). However SLIMUTIL.ASM is shown here in listing 1.

The name WATOR.PAS is used in the compile/link instructions throughout and wherever the information applies to either program. In FATFISH and SLIMFISH the pieces of code that correspond to each technique are designated by number, facilitating comparison between the two.

Before trimming, FATFISH.PAS uses Pascal runtime routines whenever possible, for example, when reading from the keyboard and performing screen I/O. It also takes advantage of Pascal's error-checking capabilities to detect out-of-range variables, stack overflows, and similar conditions. Only when a function cannot be performed in Pascal does the fat version resort to assembly language utilities.

The .ASM utilities it uses are routines that handle Ctrl-Break, permitting easy exit from the program. The techniques used are described in Tech Notebook 35 ("We Interrupt This Program," Ted Forgeron, April 1985, p. 42).

The fat version of the program can be compiled, the utilities assembled, and the entire package linked by entering the commands below. These commands assume that the source code resides in files WATOR.PAS and ASMUTIL.ASM. (FATFISH.PAS and FATUTIL.ASM must be renamed before execution.) The commands place the object code in files WATOR.OBJ and ASMUTIL.OBJ and the linked, executable code in WATOR.EXE.

```
PAS1 WATOR.PAS, WATOR.OBJ,
    WATOR.LST;
PAS2
MASM ASMUTIL.ASM, ASMUTIL.OBJ,
    ASMUTIL.LST;
LINK WATOR.OBJ + ASMUTIL.OBJ,
    WATOR.EXE, WATOR.MAP,
    PASCAL.LIB + MATH.LIB;
```

Users of IBM Pascal 2.0 or versions of MS-Pascal prior to 3.20 should omit MATH.LIB from the LINK command. IBM users instead should set up PASCAL.LIB with the EMULATOR option. MS-Pascal users need do nothing else.

With an initial code size of 34,004 bytes and a data and constants size of 65,535 bytes, FATFISH.PAS begins at a hefty 97.2KB. The accompanying table lists the number of bytes and their translation into a reduction percentage following each step. These figures are taken from WATOR.MAP, which is produced during the link step. Much of the fat in a Pascal program is allocated after the program loads. The .EXE file will *not* occupy the entire "weight" (97.2KB, for example) figure.

THAT LEAN, HUNGRY LOOK

The following weight-reduction steps resulted in a final 70 percent reduction in the size of the WATOR.PAS program.

Step 1. The first step is to turn off the runtime debugging features. FATFISH.PAS includes the metacommands $DEBUG+, $ENTRY+, and $LINE+. These features are useful, but they bloat the program's size and create a huge performance overhead.

For example, $ENTRY+ generates a call to a Pascal runtime routine each time the program calls a procedure or a function. $LINE+ calls a runtime routine for each line of the program's source code. $DEBUG+ causes runtime routines to perform additional checks when the program does arithmetic, references arrays, uses pointers, and so on.

During program development, these features enable Pascal to return error messages that list the procedure name, the source line number, and the specific error. Without debugging features, a program issues far fewer informative error messages during execution. This could cause some errors to go by unnoticed, ultimately resulting in a crash that requires a reboot.

However, once a program has been debugged, none of these errors should occur, therefore none of these features is necessary. Removing the features by replacing them with $DEBUG−, $LINE−, and $ENTRY− shrinks the program by 8.2 percent and improves performance dramatically.

Step 2. This step eliminates the code that displays error messages by linking the program to NULE6.OBJ (available on the Pascal distribution diskette). NULE6.OBJ terminates a program when an error occurs, instead of reporting errors on-screen. Removing the extra error-handling code involves a single change to the LINK command:

```
LINK WATOR.OBJ + ASMUTIL.OBJ +
   NULE6.OBJ, WATOR.EXE, WATOR.MAP,
   PASCAL.LIB + MATH.LIB;
```

Step 3. In this step, the Pascal file system is removed. FATFISH.PAS uses the file system to read characters from the keyboard (READ and READLN) and to display characters on the screen (WRITE and WRITELN). It makes screen and keyboard I/O easy, but at a great cost in code size. Instead, I/O can be performed directly. The system also provides facilities for reading from and writing to disk, which are both unnecessary to this program.

The routines added to WATOR.PAS to perform the I/O directly are designated in SLIMFISH.PAS; they include the Pascal procedures CONCAT_STRING, CONCAT_INTEGER_TO_STRING, STRING_TO_INTEGER, GET_NEXT_KEY, DISPLAY_NUMBER, GET_INTEGER, and DISPLAY_STRING.

These routines not only decrease program size, they also add features. For example, FATFISH.PAS is subject to snow on a color monitor; in the slim version, an assembly-language routine called BLAST_VIDEO_RAM synchronizes with the color/graphics adapter's vertical retrace to eliminate this distracting side effect. In addition, the slim version displays sharks in intense characters, an impossibility with the Pascal WRITE statements in FATFISH.PAS. The SLIMFISH.PAS custom I/O routines can access the attribute byte for each character on the screen.

Finally, screen I/O is much faster after this step because the program puts the images directly into the video RAM. But performing I/O in this way has its drawbacks. Because SLIMFISH.PAS accesses the video RAM directly, it is less portable. The slim version can run only on computers that assign video RAM to the IBM standard addresses of B000H for the monochrome adapter and B800H for the color/graphics adapter.

Another problem is that without special code, the hardware-generated screen cursor would flash wildly in the middle of the screen during program execution. The sharks won't eat it and the fish would swim right over it. The routines CURSOR_DISAPPEAR and CURSOR_REAPPEAR are included in SLIMUTIL.ASM to remedy the situation. The code for CURSOR_DISAPPEAR was borrowed from Tech Notebook 43 ("The Disappearing Cursor," Ted Forgeron, July 1985, p. 35).

TABLE: *Memory-requirement Reductions*

	CODE SIZE (bytes)	DATA AND CONSTANTS SIZE (bytes)	TOTAL SIZE (KB)	PERCENTAGE SAVED
Original	34,004	65,535	97.2	—
After Step 1	25,824	65,535	89.2	8.2
2	25,236	65,535	88.6	8.5
3	14,372	65,535	78	19.8
4	7,088	65,535	70.9	27
5	6,534	65,535	70.3	27.7
6	5,213	65,535	69	29
7	5,277	24,576	29.2	70

These figures are taken from WATOR.MAP, which is produced during the link step. Most of the fat in a Pascal program is allocated after the program loads. The .EXE file will not occupy the entire "weight" (97.2KB) for example.

The program must be linked to the file NULF.OBJ to keep unnecessary file system runtime routines from being linked in from the PASCAL.LIB library. The new link command is as follows:

```
LINK WATOR.OBJ + ASMUTIL.OBJ +
  NULE6.OBJ + NULF.OBJ, WATOR.EXE,
  WATOR.MAP, PASCAL.LIB + MATH.LIB;
```

After the program is recompiled and relinked with the changes thus far, its code size is reduced by 19.8 percent.

Step 4. This step removes support for real numbers. Many programs use the REAL data type because they need to represent large numbers (greater than 32,767 or less than −32,767), not because of its accompanying floating-point precision, yet they must carry the code overhead that it brings. FATFISH.PAS uses floating-point numbers in its random-number generator to provide enough range so that a reasonably random selection is produced.

The often overlooked INTEGER4 can be used instead. Variables of this type can store numbers in the range of −2,147,483,647 to 2,147,483,647, which is usually sufficient. INTEGER4 variables are used in the SLIMFISH.PAS random-number generator.

Because IBM/Microsoft Pascal supports the extensive IEEE floating-point format and can take advantage of an 8087 or 80287 if one is present, removing real numbers makes the program significantly smaller. Application of this step produces a program that is 27 percent smaller than the original. Once the program has been recompiled, it no longer needs to be linked to the MATH.LIB library. Therefore, the appropriate LINK command is:

```
LINK WATOR.OBJ + ASMUTIL.OBJ +
  NULE6.OBJ + NULF.OBJ, WATOR.EXE,
  WATOR.MAP, PASCAL.LIB;
```

Step 5. The heap is removed in step 5. In Pascal programs, variables referred to by pointer types are allocated space in an area of memory called the heap. Its removal shrinks the code but, more importantly, prepares the program for a significant reduction in step 7.

When the heap is eliminated, so is the need for HEAHQQ_CODE, a management module that is linked to any program using the heap (and is resident in PASCAL.LIB). Another benefit involves management of the data segment. With no heap, programmers can determine what the size of the data segment should be merely by calculating the size of the static data and constants, and by determining the worst-case size of the stack. Then they can shrink the data segment accordingly. Step 7 explains this in detail, but eliminating the heap facilitates the process.

FATFISH.PAS keeps a record of each fish or shark. It uses the Pascal heap-management routines NEW and DISPOSE to add or remove records. To eliminate reference to the heap, SLIMFISH.PAS statically allocates an array called FREE_POOL to hold all the FISHES records. Then it calls the custom routines ALLOCATE_FISH and FREE_FISH, which replace NEW and DISPOSE, respectively.

Instead of returning a pointer as NEW does, ALLOCATE_FISH returns an address type (ADR) of the FISHES record that stores information about a particular fish or shark. FREE_FISH adds the records for those fish or sharks that have died back into the list of free FISHES records.

This method has one drawback. It uses address types to refer to processor-specific memory addresses. This makes the program less portable than one that uses the more general pointer type. In fact, it limits the program to computers that use the base:offset form of addressing (the 8088 or other processors in Intel's iAPX 86 family). But by sacrificing portability, the program becomes smaller and faster. This step's incremental reduction is minor, but it has prepared the program for step 7.

Step 6. This step eliminates unnecessary runtime routines. To provide runtime support, the Pascal compiler always includes references to external variables and procedures defined in PASCAL.LIB. When the program is linked, the modules containing these symbols are copied from PASCAL.LIB to the program's .EXE file. However, some of the routines are not necessary to the program.

One way to discover which routines are needed is to link the program without PASCAL.LIB, then look at the link errors (unresolved externals) to see which routines the program thinks it still needs. For example, FATFISH.PAS modified through step 5 can be linked with the following command:

```
LINK WATOR.OBJ + ASMUTIL.OBJ +
  NULE6.OBJ + NULF.OBJ, WATOR.EXE,
  WATOR.MAP;
```

Several unresolved externals result:

```
TICS in file(s):
  WATOR.OBJ(WATOR)
STKHQQ in file(s):
  NULF.OBJ(NULFQQ)
LRNGQQ in file(s):
  WATOR.OBJ(WATOR)
DOSXQQ in file(s):
  WATOR.OBJ(WATOR)
SOVGQQ in file(s):
  NULF.OBJ(NULFQQ)
BEGXQQ in file(s):
  WATOR.OBJ(WATOR)
ENDXQQ in file(s):
  NULE6.OBJ(NULE6)
LDNGQQ in file(s):
  WATOR.OBJ(WATOR)
```

This output from the LINK program lists the unresolved symbols, the file in which they were referred, and the module that referred to them. In normal circumstances, the modules that define these symbols would be linked in from PASCAL.LIB. If any of these are unnecessary, dummy procedures with the same name can be substituted.

As a further determination, look at the link map of PASCAL.LIB (available on one of the Pascal distribution disks). This map lists all the public symbols defined in PASCAL.LIB, the modules that contain them, and the size of the modules. If the unresolved symbols are defined in small modules with other symbols that also are referred to, those

VT100/VT52 & Tektronix™ 4010/4014 Terminal Emulator

VTEK™ 3.0

Excellent emulation and the features you want:

- → use 4096 x 3120 resolution
- → zoom, pan, and window plots
- → high resolution printer dumps
- → choose text and plot color
- → transfer files with XMODEM and Kermit protocols
- → scroll last 4 pages of text
- → 132 column VT100 capability
- → 18 User-definable keys
- → capture plots and text on disk
- → full or half duplex
- → access to DOS commands
- → all VT100 keypad commands
- → command line editing
- → fast direct screen access
- → password security

VTEK makes your PC better than a terminal

$150 from Scientific Endeavors

Presentation Quality Graphics for Scientific and Technical Applications

GraphiC™ 2.1

- → linear, log, & polar plots
- → bar charts & Smith charts
- → contour plots with labels
- → 3-D curves, 3-D surfaces with hidden line removal
- → 4 curve types, 8 markers
- → 12 fonts, font editor
- → multiple levels of super/sub scripts
- → 4096 x 3120 resolution
- → zoom, pan, window plots
- → multiple plots on a page
- → high resolution printer dumps, full or half page
- → plotter support

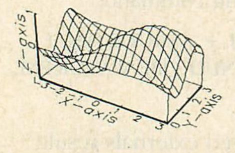
Over 100 routines can be called by your C program. SOURCE INCLUDED for private use only. $250. Demo $8.

For DeSmet, C-86, Lattice, and Microsoft C compilers.

Scientific Endeavors

Route 4, Box 79; Kingston, TN 37763

(615) 376-4146



For 256k IBM and Corona PCs, DOS 2.xx,3.xx. Epson, Okidata, Toshiba, Prowriter printers. Hewlett Packard, Houston, Sweet-P plotters. Corona Laser printer. Amdek, IBM, IBM Enhanced, Tecmar, Hercules, Corona graphics.

THIS AD WAS MADE USING GraphiC™

CIRCLE NO. 187 ON READER SERVICE CARD

TURBO EDITASM

Introducing the first co-resident editor assembler for the IBM PC family. ***TURBO EDITASM (TASM)*** is significantly faster and easier to use than the IBM Macro-Assembler (MASM). Whether you are new to assembly language and want to quickly write a small assembly language routine, or are an experienced MASM user tired of waiting months to assemble large files, ***TURBO EDITASM*** will bring the excitement back to assembly language.

***TURBO EDITASM* IS MUCH FASTER:**

- How fast is ***TASM?*** The graph below shows relative assembly times for a 48K source file. For large files like this we blow MASM's doors off at 3 times their speed. For smaller 8K files we positively vaporize them at 6 times their speed.

  TASM (110 sec.)
  MASM (340 sec.)

- ***TURBO EDITASM*** is faster for the following reasons: **(1)** Written entirely in assembly language (unlike MASM). **(2)** Editor, assembler and source file always in memory so you can go instantly from editing to assembling and back. **(3)** Eliminates the time needed to LINK programs. Executable COM files can be created directly. (Also creates OBJ files compatible with the IBM linker).

***TURBO EDITASM* IS EASIER TO USE:**

TASM includes many other features to make your programming simpler.

- Listings are sent directly to screen or printer. Assemblies can be single stepped and examined without having to leave the editor.
- Access the built-in cross reference utility from the editor.
- Full support of 186 and 286 (real mode) instructions.
- Both Microsoft and 8087 floating point formats are supported. 8087 and 287 instructions supported directly without macros for faster assembly.
- Calculator mode: Do math in any radix even using symbols from the symbol table.
- Direct to memory assembly feature lets you test execute your code from editor.
- Coming soon: A coordinated symbolic debugger.

COMPATIBILITY: *TASM* is source code compatible with MASM and supports macros, records and structures.

Introductory Price $49
With .OBJ Capability $99

Include $5.00 shipping and handling. California residents add 6% Sales Tax.

Dealer Inquires welcome

916-988-7426
118 Buck Circle, Box T
Folsom, CA 95630

Speedware™

IBM, Microsoft trademarks of IBM Corp., Microsoft Corp.

CIRCLE NO. 190 ON READER SERVICE CARD

INTRODUCING

DATA LYNX

The First Full Implementation of a

NETWORK DATA BASE

for Less Than $1000

The network model is superior to the relational and hierarchical models because it allows complex data relationships to be represented in the most natural and efficient manner.

DATA LYNX features include:

- many to many and recursive relationships —goes beyond the **CODASYL** standard
- same type of data base as found in most mainframe environments
- variable length records
- 5 levels of security
- interfaces to most popular languages
- interactive mode
- hashing
- multiuser version available

For a risk free trial or information, contact:

VANGUARD TECHNOLOGIES

3776 First Avenue, Suite A
San Diego, CA 92103

(619) 260-0007

CIRCLE NO. 193 ON READER SERVICE CARD

We will do whatever it takes to make DSD86 the best debugger available for the IBM PC.

For starters, we have by far the best design, a superior base to build from. While the competition adds new "modes" for every new feature, we have a pure, consistent and expandable design. While the competition forces you to accept their particular philosophy, we offer maximized flexability. If you already have a debugger or are looking for your first, look no further because you can't do any better. We invite you to compare our debugger, DSD86, with any other on the market; let the loser refund your money.

- Recursive Command Macros and Command Files
- Bind Macros or Commands to any Keys
- Instructions, Registers and Stack Displayed Symbolically
- Customize Screen Layout with built in Windowing System
- "Mode-less Design - All Commands Available at All Times
- Source Level Debugging for Microsoft Languages
- User Writeable Displays and Commands
- 30 Day Money Back Guarantee

An incredible value for only $69.95!

Soft Advances

P.O. Box 49473 Austin, TX 78765

512-478-4763

Please include $4 for shipping & handling.

CIRCLE NO. 176 ON READER SERVICE CARD

modules can be linked in with little size bloat. However, if an unresolved symbol is defined in a large module with other symbols that are not referred to, that symbol may be excluded.

Examination of the link map shows that the unresolved symbols from WATOR.PAS are defined in the following PASCAL.LIB modules:

```
TICS    from TIDGQQ_CODE
STKHQQ from ENTX6L (ENTX6S for IBM)
LRNGQQ from LONGQQ
DOSXQQ from ENTX6L (ENTX6S for IBM)
SOVGQQ from MISGQQ
BEGXQQ from ENTX6L (ENTX6S for IBM)
ENDXQQ from ENTX6L (ENTX6S for IBM)
LNDGQQ from LONGQQ
```

At least one of these modules, ENTX6L (or ENTX6S) is required. It performs many important tasks such as initializing global variables and configuring runtime memory. This module will be used in step 7; its source code is contained on the Pascal distribution disk. To be certain it does not refer to other symbols in PASCAL.LIB, it is assembled and linked to the program:

```
MASM ENTX6L.ASM, ENTX6L.OBJ,
   ENTX6L.LST;
LINK WATOR.OBJ + ASMUTIL.OBJ +
   NULE6.OBJ + NULF.OBJ +
   ENTX6L.OBJ, WATOR.EXE, WATOR.MAP;
```

Although this resolved the references to the symbols defined in the module ENTX6L (or ENTX6S), two new unresolved external references appeared. The updated list of symbols and the modules that contain them is as follows:

```
TICS    from TIDGQQ_CODE
LRNGQQ from LONGQQ
SOVGQQ from MISGQQ
LDNGQQ from LONGQQ
BEGOQQ from MISOQQ_CODE
ENDOQQ from MISOQQ_CODE
```

But it must be determined whether a procedure performs a function necessary to the program or if it is being linked in because of a wasteful compiler convention. Sometimes the function can be discerned from the symbol or module name. If more information is needed, recompile the program and include a file name to receive the generated assembly-language code. Then run PAS3 to produce the code listing.

The listing will include references to the unresolved symbols. It also will show the associated line number of the original Pascal source program. This information is generally helpful to the experienced programmer in determining what PASCAL.LIB procedures do.

The next step is to determine the importance of the unresolved symbols for this program. LRNGQQ and LDNGQQ perform important functions. They contain the runtime code necessary to support INTEGER4 variables. The module TIDGQQ_CODE contains several routines for manipulating the date and the time, but the sample program needs only one of them, TICS.

TICS simply reads the system timer. This program uses it to seed the random-number generator. Instead of linking in the entire TIDGQQ_CODE module, SLIMFISH.PAS includes its own version of TICS (the code for which is included in SLIMUTIL.ASM).

The link message declaring SOVGQQ an unresolved external implies that it is needed by the file system, NULFQQ. But SLIMFISH.PAS is no longer using the system, and NULFQQ is supposed to provide its own dummy references to resolve all the external references to the file system. This sounds like a symbol that slipped by the null file system. SLIMFISH.PAS can include a dummy symbol for SOVGQQ and save 876 bytes.

BEGOQQ and ENDOQQ perform program initialization and termination functions unnecessary to WATOR.PAS. SLIMFISH.PAS includes dummy symbols for the two and saves 59 bytes.

To establish the dummy variables, SLIMFISH uses a file called DUMMY.ASM (listing 2) that satisfies external symbols for the linker. When the routines in DUMMY.ASM are called, they return with no action. Alterations through step 6 produce a program that is 29 percent smaller than its original. The new assemble, compile, and link sequence is as follows:

```
PAS1 WATOR.PAS, WATOR.OBJ,
   WATOR.LST;
PAS2
MASM ASMUTIL.ASM, ASMUTIL.OBJ,
   ASMUTIL.LST
MASM DUMMY.ASM, DUMMY.OBJ,
   DUMMY.LST;
LINK WATOR.OBJ + ASMUTIL.OBJ +
   NULE6.OBJ + NULF.OBJ +
   DUMMY.OBJ, WATOR.EXE,
   WATOR.MAP, PASCAL.LIB;
```

Step 7. This final step limits the size of the stack and the heap. Pascal runtime control routines set up a program's data segment in a way that can be extremely wasteful for small programs. It checks the amount of free memory in the computer and grabs as much as it can, up to 64KB, for the data segment. Then it moves all the constants and static variables up to the highest addresses of that segment and lets the heap grow from the bottom and the stack grow down from the end of the static data.

This gives the stack and the heap a lot of room to grow, minimizing overflow problems, but it also means that even the smallest program *is stuck with* a huge 64KB data segment. The source code for the routines that perform this manipulation is supplied on the Pascal distribution diskettes. With some simple modifications to either ENTX6L.ASM (for MS-Pascal) or ENTX6S.ASM (for IBM), Pascal programs can set the maximum size of their own data segments. (These modifications were suggested by Blaise Computing, Inc., a company that produces Pascal tools.) To modify ENTX6S.ASM or ENTX6L.ASM, the first step is to find the data declarations labeled with the comment:

```
; System resident public data
```

Right before the DATA ENDS statement, add the following line:

```
extrn datsqq:word      ;*Data segment size in
   paragraphs
```

Now find the statement labeled:

```
BEGXQQ     PROC     FAR
```

Move down 16 or 17 instructions to:

```
SUB AX,HIMEM ;this is the number of para's
   available total
```

Immediately after that line, add the following instruction:

```
mov ax,datsqq ;*this is number of
                 available paragraphs,
              ;*and will force a data
                 space of approx
              ;datsqq
```

That is the extent of the necessary modifications. This article endorses Blaise's suggestion of storing this modified file as ENTXSML.ASM and leaving the original as is.

To supply the information on the size of the data segment, the program must declare DATSQQ as a public variable and use a value statement to initialize its value to the number of paragraphs (16-byte blocks) in the data segment. (See SLIMFISH.PAS for the formatting of these statements.)

The next problem is to decide how big the data segment should be. Step 5 provided a partial solution. It removed the heap from consideration by setting up a static array to contain all the fish records. The size of all this static data can be determined by looking at the link map and subtracting the start ad-

dress for HEAP from the stop address for HIMEM, leaving only the size of the stack in question.

Determining the size of the stack is a matter of judgment (and possibly trial and error, if the program crashes later). For this program, 2KB of stack is very generous since the program does not contain recursive routines, nor does it pass much data on the stack. Therefore, by adding up all the data and constants listed in the link map, and by adding 2KB for the stack, the real requirement for the data segment is 24,576 bytes. Then DATSQQ can be set accordingly. (The DATSQQ value is assumed to be the number of 16-byte paragraphs, so it is 24,576 divided by 16 or 1,536.)

To make the changes that are necessary to this step, modify WATOR.PAS to add DATSQQ and recompile the file. Next, create ENTXSML.ASM and assemble it, then link all the necessary files together. The complete set of commands necessary to assemble, compile, and link this final slim version is as follows:

```
PAS1 WATOR.PAS, WATOR.OBJ,
   WATOR.LST;
PAS2
MASM ASMUTIL.ASM, ASMUTIL.OBJ,
   ASMUTIL.LST;
MASM DUMMY.ASM, DUMMY.OBJ,
   DUMMY.LST;
MASM ENTXSML.ASM, ENTXSML.OBJ,
   ENTXSML.LST;
LINK WATOR.OBJ + ASMUTIL.ASM +
   NULE6.OBJ + NULF.OBJ +
   DUMMY.OBJ + ENTXSML.OBJ,
   WATOR.EXE, WATOR.MAP,
   PASCAL.LIB;
```

The result is SLIMFISH.PAS, a program that requires 70 percent less memory than FATFISH.PAS, its former self. It is interesting to note the slim version contains more source code. In addition, SLIMFISH runs considerably faster. With the recommended values set, 10 generations of FATFISH take 384 seconds to run, while the same number of generations of SLIMFISH take only 13 seconds. (These times represent an averaging of several runs.)

The only thing that could make this program substantially smaller and faster would be to recode it entirely in assembly language. But with a code size that is already down to 5,277 bytes, it is probably not worth the effort.

Steven Armbrust is a freelance technical writer. Ted Forgeron is a microcomputer software consultant. They work primarily in the "Silicon Forest" near Portland, Oregon.

THE TOROIDAL PLANET OF WA-TOR

The program used in this article is a game that was discussed in "Computer Recreations" of *Scientific American* (A. K. Dewdney, December 1984, p. 14. The algorithms used in the *PC Tech Journal* version were inspired by Jay Jaeger of the Wisconsin Department of Transportation.) The program WATOR simulates the toroidal (doughnut-shaped) world of Wa-Tor, a liquid-covered planet where the two dominant forms of life resemble fish and sharks.

The fish thrive on an unending supply of plankton that fills Wa-Tor's planetary sea. They reproduce at regular intervals defined by the player during the set-up phase of the program. Sharks stay alive by feeding on the fish; the length of time they can survive without starving and the intervals at which they reproduce are defined by the player during setup. After setup is complete, life on Wa-Tor takes over the screen.

Fish and sharks spring into existence, living and dying by the rules just established. In some scenarios the sharks are unable to find food nearby, or they cannot breed fast enough, and they slowly die while the fish take over the world. In other scenarios the sharks reproduce too quickly (or the fish not quickly enough), and they feast greedily on the entire fish population, dooming themselves to certain extinction. In some delicately balanced scenarios the fish and shark populations rise and fall in sinusoidal patterns, similar to the predator/prey relationships that occur in real life.

Both versions of this program (which are available for downloading on PCTECHline) run on monochrome or color/graphics screens. To run either of them, follow the compiling and linking instructions listed in this article. Then type:

```
WATOR
```

The slim version (SLIMFISH.PAS) runs much faster and includes features such as high-intensity characters that are not possible with standard Pascal.

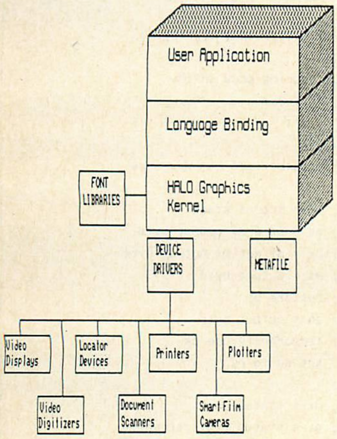
—*SA and TF*

LISTING 1: SLIMUTIL.ASM

```
        TITLE   ASMUTIL - Assembly language utilities module

TRUE            EQU     01H         ; boolean true
FALSE           EQU     00H         ; boolean false
STCRSR          EQU     02H         ; bios set cursor
SCROLLUP        EQU     06H         ; bios scroll page
VIDEO_IO        EQU     10H         ; bios video i/o
EQUIP           EQU     11H         ; bios equipment selections
SET_CURSOR_TYPE EQU     01H         ; bios cursor type option
BREAKINT        EQU     1BH         ; bios ctrl-break interrupt
SYS_TIMER       EQU     1AH         ; bios time of day interrupt
DOS_FUNCTION    EQU     21H         ; dos function call
GETVECTOR       EQU     35H         ; dos get vector function
SETVECTOR       EQU     25H         ; dos set vector function
STATUS_PORT     EQU     3DAH        ; status port address

ASMUTIL SEGMENT PUBLIC 'CODE'

        ASSUME  CS:ASMUTIL
        ASSUME  DS:NOTHING

BREAKFLAG       DB      0           ; break key hit flag
SAVEBREAK       DD      0           ; saved copy of break vect.

;
; function tics: word;
;
; Returns the low word of the system clock.
;
        PUBLIC  TICS
TICS            PROC FAR
        MOV     AH,0                ; read clock setting
        INT     SYS_TIMER           ; bios time of day
        MOV     AX,DX               ; return value in ax
        RET
TICS ENDP
;
; function equipment: word;
;
; Get current bios equipment selections.
;
        PUBLIC  EQUIPMENT
EQUIPMENT       PROC FAR            ; return bios equipment list
        INT     EQUIP               ; bios equipment routine
        RET                         ; equip list returned in ax
EQUIPMENT       ENDP
;
; procedure set_cursor(row, column: integer);
;
; Position cursor on screen.  Assumes video page 0.
;
        PUBLIC  SET_CURSOR
SET_CURSOR      PROC    FAR         ; set cursor on text page
                                    ; with row and col on stack
        PUSH    BP                  ; save frame pointer
        MOV     BP, SP              ; get stack top
        MOV     DH, [BP+8]          ; set dh = row
        MOV     DL, [BP+6]          ; set dl = column
        MOV     BH, 0               ; default to page zero
        MOV     AH,STCRSR           ; set ah = set cursor
        INT     VIDEO_IO            ; call bios video i/o
        POP     BP                  ; restore frame pointer
        RET     4
SET_CURSOR ENDP
;
; procedure cursor_disappear
;
; Make the cursor invisible.
;
        PUBLIC  CURSOR_DISAPPEAR
CURSOR_DISAPPEAR PROC FAR
        PUSH    AX                  ; save registers
        PUSH    CX
        MOV     CH,00100000B        ; bit 5 on, bit 6 off
        MOV     AH,SET_CURSOR_TYPE  ; set cursor type option
        INT     VIDEO_IO            ; call bios video i/o
```

HALO the market standard for graphics

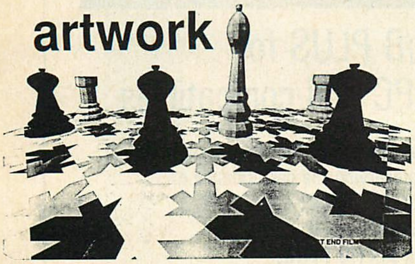
HALO provides software implementors with a complete graphics development environment.



Device Independent

Each copy of HALO comes with a full complement of display, input, printer, film recorder and plotter device drivers. Metafile creation and display are included. Media Cybernetics is continually developing new device drivers to meet the ever changing new hardware technologies.

HALO supports all IBM compatible MS/PC DOS systems including Texas Instruments Professional™, AT&T 6300™, IBM™ 3270-PC and of course the IBM PC, XT and AT. Over twenty graphics display boards manufactured by IBM and leading third party vendors are supported. A single device driver supports all display modes on each device. Image input from video digitizers and document scanners is supported. New device drivers include the IBM EGA and Professional and the AT&T Image Capture and Display boards.



This picture was created with "Artwork", a HALO-based application written by West End Film of Washington, DC.

De-Facto Standard

First released in 1982, HALO has an established base of over 40,000 end users and over 100 corporate clients. Numerous HALO-based CAD, solids modeling, presentation graphics, art packages, mapping and other vertical market software applications are commercially available.

As early as June 1984, PC World featured an article entitled, "HALO A new software library leads the way toward graphics standardization and portability." The July 9, 1985 issue of PC Week" featured users stories from both Rockwell International Corp., and Lawrence Livermore Labs on how they use HALO to save development time and money. Most recently, Mini-Micro Systems August 1985 issue stated, "Widely used, HALO has attained de facto-standard status. It's certainly the most widely used library." HALO has achieved this status because it provides a complete device independent graphics environment for software developers. Since the HALO interface rarely changes, compatibility with a new device is achieved simply by adding a new device driver.

By incorporating all the HALO device drivers into their applications, licensed commercial software developers solve distribution and compatibility problems.

This picture was created with "Design Board 3-D", a HALO-based application written by MEGA-CADD of Seattle, Washington.

Powerful Graphics Toolbox

HALO has over 170 functions. Standard functions include: point, line, arc, circle, ellipse, polygon, etc. Coloring control functions are available for color selection, palette management, dithering, and textures.

Advanced features encompass: image compression routines for reducing data storage requirements; both absolute and relative addressing plus individual cursors for graphics text and interaction; world and normalized coordinate systems for windowing, viewport and clipping; and a metafile reader and generator.

HALO's raster extension supports rubberbanding and non-destructive modes so that polygons and text may be interactively created and moved over a displayed image.

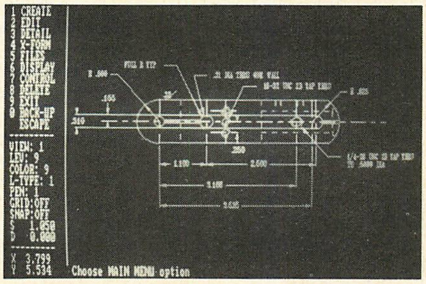
There are currently over twenty bit mapped and stroke text fonts available for HALO. The object oriented type faces provide user definition of line width, size, proportional width, angle, and interior fill style.

HALO's Virtual Rasterization Interface (VRI) frees users from the limited resolution of their graphics board by modeling the graphics display in memory and having the hardcopy produced in the maximum resolution of the output device.

HALO can be addressed as a linkable library or as a resident driver depending on the application.

Faster than VDI

In benchmark tests by independent companies evaluating HALO and GKS, HALO was determined to be three to ten times faster depending on the function tested. These benchmark results made HALO a requirement for many government agencies and large corporations. This gives HALO compatible hardware and software significant inroads into these agencies and corporations.



This picture was created with "CADKEY", a HALO-based application written by MICRO CONTROL SYSTEMS of Vernon, Connecticut.

Easy to Learn

HALO comes with a computer aided tutorial. This tutorial, entitled LEARNHALO, teaches users to become proficient graphics developers in less than a day by providing step-by-step instructions through graded examples.

HALO is written in Assembler and callable from the language you are already comfortable with: Interpretive and Compiled Basic, Professional Fortran, MS Fortran™, Pascal, Turbo Pascal™, MS C™, Mark Williams C™, Lattice C™, Aztec C™, Computer Innovations C86™, and Golden Common Lisp™.

Increase Market Value of Hardware

Graphics hardware manufactures receive distinct marketing advantages by having their products supported by HALO. HALO is the gateway to an extensive body of commercially available software.

Pricing Information

A single copy of HALO for one language costs $250.00. A second language binding ordered at the same time is an additional $150.00. Author, Corporate and Site licensing and OEM agreements are negotiated on a case by case basis.

To order, or receive more information on HALO and other Media Cybernetics, Inc., products contact:

media cybernetics, inc.
7050 Carroll Avenue
Takoma Park, MD 20912
(800) 446-HALO or (301) 270-0240
Telex #332014

HALO, LEARNHALO, Texas Instruments Professional, AT&T, IBM, MS Fortran, Microsoft C, Turbo Pascal, Mark Williams C, Lattice C, Aztec C, Computer Innovations C86, Golden Common Lisp and GKS VDI are all registered trademarks of Media Cybernetics Inc., International Business Machines, Texas Instruments Inc., AT&T, Microsoft Corp., Borland Intl., Mark Williams Co., Lattice Inc., Manx Software Systems, Computer Innovations Inc., Gold Hill Computers, and Graphics Software Systems Inc., respectively.

CIRCLE NO. 204 ON READER SERVICE CARD

```
        POP     CX                  ; restore registers
        POP     AX
        RET
CURSOR_DISAPPEAR ENDP
;
; procedure cursor_reappear
;
; Makes the normal flat cursor reappear on the screen.
;
        PUBLIC  CURSOR_REAPPEAR
CURSOR_REAPPEAR PROC FAR
        PUSH    AX                  ; save registers
        PUSH    CX
        MOV     CH,0BH              ; cursor start reg for mono
        MOV     CL,0CH              ; cursor end reg for mono
        CALL    EQUIPMENT           ; ax = bios equipment flags
        AND     AX,30H              ; mask off mono bits
        CMP     AX,30H              ; is this a mono adapter?
        JE      C1                  ; yes, keep mono start & end
        MOV     CH,06H              ; cursor start reg for color
        MOV     CL,07H              ; cursor end reg for color
C1:     MOV     AH,SET_CURSOR_TYPE  ; set cursor type option
        INT     VIDEO_IO            ; call bios video i/o
        POP     CX                  ; restore registers
        POP     AX
        RET
CURSOR_REAPPEAR ENDP
;
; procedure clear_screen;
;
; Clear video screen.
;
        PUBLIC  CLEAR_SCREEN
CLEAR_SCREEN    PROC FAR             ; clear screen
        PUSH    BP                   ; save frame pointer
        MOV     BP, SP               ; get stack top
        MOV     AL,0                 ; blank entire window
        MOV     CH,0                 ; row of upper left corner
        MOV     CL,0                 ; col or upper left corner
        MOV     DH,24                ; row of lower right corner
        MOV     DL,79                ; col of lower right corner
        MOV     BH,7                 ; normal attributes
        MOV     AH,SCROLLUP          ; AH = scroll active page up
        INT     VIDEO_IO             ; call bios video i/o
        POP     BP                   ; restore frame pointer
        RET
CLEAR_SCREEN    ENDP
;
; procedure blast_video_ram(addr:ads of byte;output_byte:byte);
;
; Blasts a byte into video ram without causing snow on the
; color/graphics adapter.
;
        PUBLIC  BLAST_VIDEO_RAM
BLAST_VIDEO_RAM PROC FAR             ; update video ram w/o snow
        PUSH    BP
        MOV     BP,SP                ; bp = top of stack
        MOV     ES,10[BP]            ; es = base of output byte
        MOV     BX,8[BP]             ; bx = offset of output byte
        MOV     AX,6[BP]             ; ax = output byte
        POP     BP                   ; restore bp
        XCHG    AH,AL                ; ah = output byte
        PUSH    DS                   ; temporarily use ds
        PUSH    ES                   ; set ds to es
        POP     DS                   ;
        MOV     DX, STATUS_PORT      ; dx = status port
SYNC1:  IN      AL, DX               ; al = status register value
        TEST    AL, 1                ; test low bit
        JNZ     SYNC1                ; if not 0 then try again
SYNC2:  IN      AL, DX               ; al = status register value
        TEST    AL, 1                ; test low bit
        JZ      SYNC2                ; if 0 then try again
        MOV     [BX],AH              ; blast byte in video ram
        POP     DS                   ;
        RET     6                    ;
BLAST_VIDEO_RAM ENDP
;
; function check_break: boolean;
```

```
;
; Checks if ctrl-break has been pressed.  It returns TRUE
; if ctrl-break has been pressed and FALSE if it hasn't.
;
        PUBLIC  CHECK_BREAK
CHECK_BREAK     PROC FAR
        XOR     AX,AX               ; clear ax
        MOV     AL, BREAKFLAG       ; return value = breakflag
        MOV     BREAKFLAG, FALSE    ; reset break flag
        RET
CHECK_BREAK     ENDP
;
; procedure install_break_handler;
;
; Installs a ctrl-break interrupt handler.  It also
; saves the address of the former break handler.
;
        PUBLIC  INSTALL_BREAK_HANDLER
INSTALL_BREAK_HANDLER PROC FAR
        PUSH    DS
        MOV     AL,BREAKINT         ; al = dos break interrupt
        MOV     AH,GETVECTOR        ; ah = dos get vector funct.
        INT     DOS_FUNCTION        ; call dos
        MOV     WORD PTR SAVEBREAK,BX   ; save off. of int. vec.
        MOV     WORD PTR SAVEBREAK+2,ES ; save base of int. vec.
        MOV     AL,BREAKINT         ; al = dos break interrupt
        MOV     AH,SETVECTOR        ; ah = dos set vector funct.
        MOV     DX,OFFSET BREAK_HANDLER ; dx = off. break hndlr
        MOV     BX,CS               ; bx = this segment
        MOV     DS,BX               ; ds = this segment
        INT     DOS_FUNCTION        ; call dos
        POP     DS
        RET
INSTALL_BREAK_HANDLER  ENDP
;
; interrupt handler break_handler;
;
; This is invoked by the bios when ctrl_break is pressed.
;
BREAK_HANDLER   PROC FAR
        MOV     BREAKFLAG, TRUE     ; breakflag = ctrl-break
        IRET                        ;  was pressed
BREAK_HANDLER   ENDP
;
; procedure remove_break_handler;
;
; Restores the previous ctrl-break handler.
;
        PUBLIC  REMOVE_BREAK_HANDLER
REMOVE_BREAK_HANDLER PROC FAR
        PUSH    DS
        MOV     AL,BREAKINT         ; al = dos break interrupt
        MOV     AH,SETVECTOR        ; ah = dos set vector funct.
        MOV     DX,WORD PTR SAVEBREAK   ; dx = saved offset
        MOV     BX,WORD PTR SAVEBREAK+2 ; bx = saved base
        MOV     DS,BX               ; ds = saved base
        INT     DOS_FUNCTION        ; call dos
        POP     DS
        RET
REMOVE_BREAK_HANDLER ENDP
ASMUTIL ENDS
        END
```

LISTING 2: DUMMY.ASM

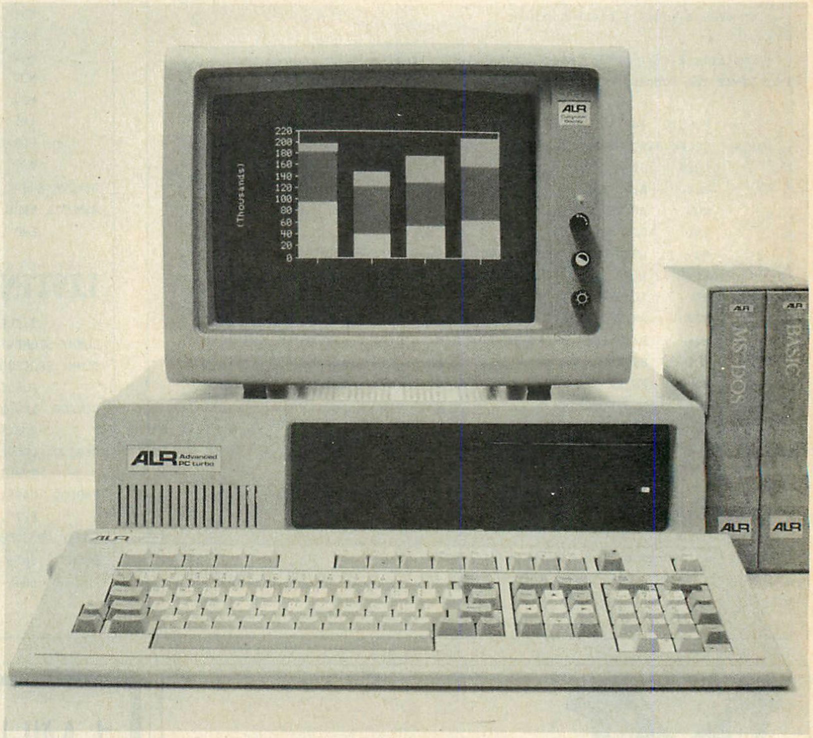
```
        TITLE   DUMMY - Dummy version of unneeded Pascal runtime
DUMMY SEGMENT PUBLIC 'CODE'
DUMMY_PROCEDURE   PROC FAR
        PUBLIC  SOVGQQ
SOVGQQ  LABEL FAR
        PUBLIC  BEGOQQ
BEGOQQ  LABEL FAR
        PUBLIC  ENDOQQ
ENDOQQ  LABEL FAR
        RET
DUMMY_PROCEDURE   ENDP
DUMMY   ENDS
        END
```

DEFINING THE IDEAL PC

com • pat • i • ble/kəm-pat-'ə-bəl/1: capable of forming a homogeneous mixture that neither separates nor is altered by electronic interaction 2: capable of operating thousands of programs utilizing the technology of the 8088-2 microprocessor 3: having the ability to run IBM PC and IBM PC/XT programs 4: having the option of exchanging data with the 1.2 Mb IBM PC AT floppy

com • pet • i • tive/kəm-pet-'ət-iv/ 1: inclined, able, or suited to compete 2: competing at **60% faster** with **8 MHz/4.7 MHz**

com • plete/kəm-plet'/1: having all necessary parts, elements or steps 2: leaving little need for add-ons 3: having **20 megabytes** storage and **640k** of memory 4: including Clock calendar, printer and serial ports 5: an enchanced keyboard with separate numeric and curser controls as standard equipment.



con • fi • dence/kan-'fəd-ən(t)s/1: the quality or state of being certain 2: the assurance of a THIRTY-DAY MONEY BACK GUARANTEE 3: having a full ONE YEAR WARRANTY on an entire system

con • clu • sion/kan-'klu-zhan/1: to reach as a logically necessary end by reasoning 2: ideal PC, ALR Advance PC turbo at $1,995 a compatible, competitive, complete system plus the confidence of the ALR guarantee.

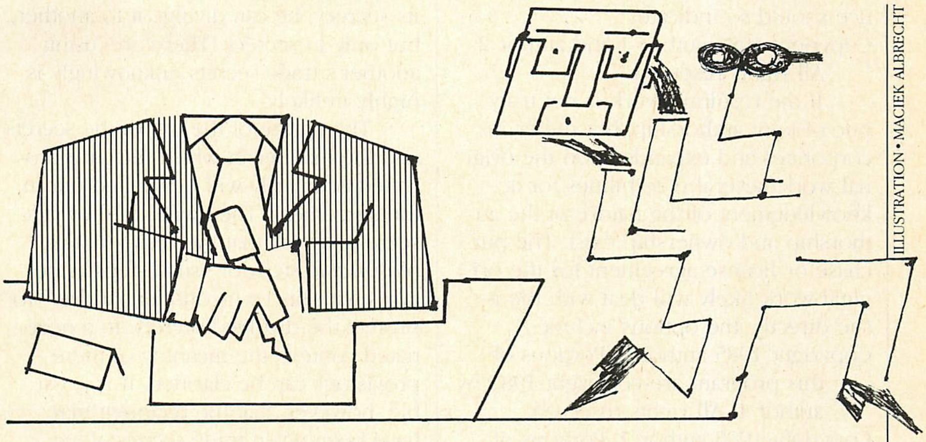
con • tact/kan'/takt/1: an establishing of communication with ADVANCED LOGIC RESEARCH, INC. by dialing **(714) 832-7808** on any standard telephone.

ALR

Advanced Logic Research, Inc.
15455 Red Hill Ave., Suite B, Tustin, CA 92680
(714) 832-7808

Acknowledging Ownership

Using or building upon other companies' products demands that appropriate credit be given to the original developer.



A distinguishing feature of the computer industry is its dependence on *intellectual property* or property that is predominantly the product of someone's thought process rather than a mechanical operation. Such property is protected differently than physical property, primarily by copyright, trademark, trade secret, and patent. All of these means have been discussed in previous columns from the viewpoint of the owner of the intellectual property.

Frequently, however, computer products depend on intellectual property that belongs to someone else: a computer manufacturer incorporates proprietary chips produced by another manufacturer; a software author compiles a program using someone else's compiler or writes an application for another database program.

Assuming that proper permission has been obtained for the use of the underlying property, attention still should be given to protecting the other party's intellectual property interests. Often the sales or licensing agreement by which the underlying property is obtained will spell out the steps to be taken. Even if not, however, these steps are usually simple and inexpensive and courtesy, if nothing else, dictates that they be taken.

In the case of a trademark, rights are obtained by using a mark to identify goods or services. Assuming that the identifier is not scandalous or purely descriptive (or otherwise prohibited by the statute in question), it belongs to the first user in commerce. Trademark rights may be lost, however, if the identifier subsequently becomes generic. The term *aspirin* is a classic case; originally it was a trademark.

Simply because a company has a trademark does not prohibit others from using it; the law merely prohibits use of the mark as a trademark for similar goods. A peripheral can be advertised as IBM-compatible. In addition, a competitor can make certain uses of a trademark—for example, "Doctors prefer Apples to IBM PCs two to one."

The owner of the trademark does, however, have a legitimate interest in being sure that its use at least identifies the trademark as a trademark, for fear that continued use by others in a generic sense may weaken and ultimately destroy the trademark. IBM would probably prefer the example to read, "Doctors prefer IBM PC® computers to Apple® computers four to one. IBM PC is a registered trademark of International Business Machines Corporation. Apple is a registered trademark of Apple Computer Corporation."

The revised example identifies the trademark by the symbol ®, indicating that it has been registered with the United States Patent and Trademark Office, and uses the trademark as an adjective rather than as a noun, both of which reduce the argument that the mark is being used in a generic sense. (The symbol TM is typically used for claimed, but unregistered, marks.) In advertising either compatibility or comparison, or even contempt, another company's trademark should be acknowledged as such.

A more difficult situation is in the use of another's copyrighted work. A valid copyright gives the author the exclusive right to reproduce the work and to prepare derivative works. The issue of when one work is original rather than a derivative of another is particularly complicated in the case of computer programs. Recent cases have begun to test the borderline between original use of another's ideas and unlawful creation of derivative works (or copying). Circumstances exist, however, when a person clearly does have the right to use (for example, by license) or build on someone else's program (for example, by writing a template for a program that a customer has independently obtained the right to use).

Two problems must be avoided: infringement or contributory infringement of the underlying work; and failure to give appropriate credit to the author of the underlying work.

Although problems of proof may be difficult, it is not a copyright infringement to create independently a work similar to the copyrighted one. Theoretically, it is possible to write a line-for-line identical version of a program and not violate its owner's copyright—convincing a court of the originality is the difficult part. In the case of an independently created program, the author simply places his own copyright notice on the product and does not worry about third parties' rights.

Some programs are sold or licensed with the understanding that the purchaser may modify them to suit his needs. In this case, depending on the degree of modification, the purchaser may have some copyright of his own. Assuming he has the right to market the resulting product, he should acknowledge the copyright of the owner of the underlying program. Again, depending on the degree of modification and the specific agreement between the original author and the author of the modifications, they may be joint authors of the finished product, and the copyright notice should so indicate:

Copyright 1985 author 1 and author 2. All rights reserved.

If the resulting work is not truly one of joint authorship, but rather incorporates and expands upon the original work, several possibilities for acknowledgment of the nature of the authorship and ownership exist. The purchase or license agreement for the original work likely will deal with the issue directly; the options include:

Copyright 1985 author 2. Portions of this program are copyright 1982 by author 1. All rights reserved.

Copyright 1985 author 2. Portions of this program are copyright 1982 by author 1 and licensed to author 2. All rights reserved. This program may not be used except in accordance with author 1's license agreement.

Coyright 1985 author 2. Copyright 1982 author 1. All rights reserved.

A different set of intellectual property issues is presented when using trade secrets that belong to another. By definition, it is almost impossible to be using someone else's trade secrets unless a specific agreement grants this right. This is because a trade secret exists only as long as its owner maintains its secrecy: he can divulge it to another, but only in secrecy. Therefore, using another's trade secrets unknowingly is highly unlikely.

The nature of the use of the secrets and the manner in which they are protected ordinarily will be spelled out in the agreement. The owner of the trade secret may ask that the recipient take *reasonable* steps or "such steps as he takes with his own confidential data" to protect the divulged secrets. In a negotiated context, the meaning of these provisions can be clarified. It is possible, however, that the recipient may have been given trade secrets along with a boilerplate trade secret agreement and may have no confidential data of his own to use as a measuring rod. In this case, an appropriate combination of the following steps is recommended in considering the form and value of the confidential information:

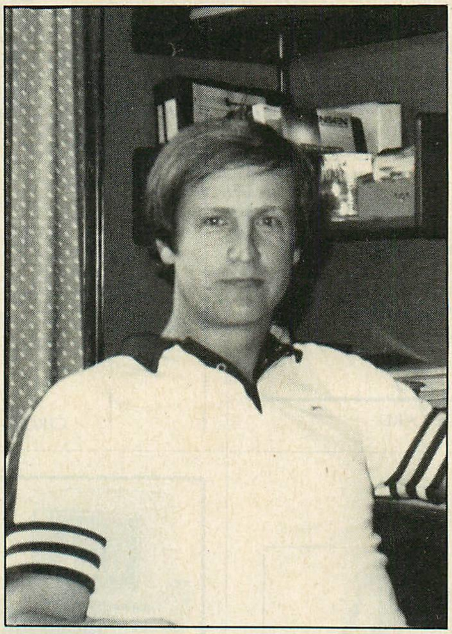
1. Keep the physical embodiment of the confidential information in a controllable location.
2. Keep track of which individuals have had access to the information.
3. Have those with access sign a confidentiality agreement similar in content to the original.
4. Limit access to the extent practical.
5. To the extent practical, keep the confidential information and the means of disseminating it separate.
6. If misuse of the information comes to the attention of the recipient, he should take appropriate action.
7. If the same or similar information comes into the recipient's possession through unrestricted channels, he should keep track of it; a trade secret loses its protection once it becomes public knowledge without the fault of the recipient.

Max Stul Oppenheimer, PC, is a partner in the law firm of Venable, Baetjer & Howard located in Baltimore.

"I Program In BetterBASIC And I Recommend It."

"I've been programming for about eight or nine years in BASIC and I'm really impressed with BetterBASIC. It's got expanded memory, windowing and extensibility. I like the cleverness of the way it's been constructed using workspaces, modules, functions and procedures. Right now I'm working on three custom business applications and all three are being written in Better-BASIC because of all the things it can do. I program in BetterBASIC and I recommend it."

Bob Briggs - Software Consultant - North Easton, MA.



640K Now you can use the full memory of your PC to develop large programs.
STRUCTURED Create well organized programs using procedures and functions that are easily identified and understood and completely reusable in future programs.
MODULAR Use procedures and functions grouped together to form "library modules" which are then available to you or anyone else for future use.
EXTENSIBLE Create your own BetterBASIC modules which contain BetterBASIC extensions. This feature coupled with the easy-to-use Assembly Language support, makes this an ideal OEM language.
INTERACTIVE BetterBASIC acts like an interpreter because it responds to the users' commands in an immediate mode. However each statement is actually compiled as it is entered.
COMPILED Each line of the program is compiled as it is entered into the computer's memory rather than interpreted at runtime.
RUNTIME SYSTEM The optional Runtime System generates stand alone EXE. files allowing for the distribution of products written in BetterBASIC with no royalties.
SUPPORTS Windows, Graphics, DOS and BIOS ROM calls, Chaining, Overlays, Local and Global Variables, Recursion . . . and more.
BetterBASIC Runs on IBM PC, XT, AT and all IBM-compatibles. Ask your local dealer for BetterBASIC or call 1-800-225-5800 in Canada call 416-469-5244. Also available for the Tandy 1000, 1200, AND 2000 AT Tandy/Radio Shack stores.

Summit Software Technology, Inc.™
P.O. Box 99, Babson Park
Wellesley, MA 02157

PRICES:

BetterBASIC	$199
8087/80287 Math Module	$99
BTrieve™ Interface	$99
Runtime System	$250
Sample Disk with Tutorial	$10

Better BASIC.

Because It's The Best.

MasterCard, Visa, Checks, Money Order, C.O.D. accepted and P.O. on approval.

(If you're using BetterBASIC and would like to be featured in one of our ads, please write to the Director of Advertising at Summit.)

CIRCLE NO. 196 ON READER SERVICE CARD

Realtime for the First Time

The important aspects of a difficult subject are given brief, but pointed treatment in this introductory text.

Introduction to Real-time Software Design
S. T. Allworth (Springer-Verlag New York, Inc., New York, 1981)
140 pages, paper, $14.95

Realtime systems are designed to create a sense of harmony between users and software and to gracefully accommodate disharmony when it inevitably occurs. They are inordinately tied to time-critical applications in which a late reply or a missed cue could cause serious problems. Realtime development therefore gives great attention to the fundamentals of operating system theory and practice, hardware and software reliability, performance, and, consequently, design methodologies. This is a decidedly complex, challenging area.

As an introductory book on the subject, Allworth's contribution is not easy to read. The vocabulary is somewhat steep and the prose is extremely efficient; this is a book of black and whites. Included is a three-page list of selected references—a big hint that a lot is being left to the reader. At only 140 pages, this book is too condensed to be successful as an introduction to anything except what is *important* in the field of realtime system development. (It is actually an aggregation and polishing of material that was presented in 20 lectures to undergraduate electrical engineers. Its brevity betrays this outline-type origin.)

Introduction to Real-time Software Design can be recommended to two groups: engineers who understand operating systems but who have not worked in a realtime situation, and computer science students who may find it similar to those condensed analysis booklets for literature. The book is intended as a reference tool; however, some of the material is not given adequate treatment even in an acknowledged abbreviated form.

The introduction previews all the topics covered. The author introduces the realtime system as a controlling element for one or more time-critical events. He discusses the fact that batch accounting is different from a realtime system (which is not organized to optimize the use of computing facilities). The importance of correctness, completeness, and reliability is emphasized. Included is a look at the cost of an example software system over its lifetime, and some attention is paid to code complexity and structure.

One-half of the book is used to discuss the features of realtime operating systems, such as multitasking, communication among simultaneously running programs, synchronization, deadlock among concurrent programs, and interrupts. Within this half, approximately equal sections are devoted to design and construction, reliability, and performance considerations.

Realtime software may be the most difficult to write, but the payoff is a complete and well-done product, much like a house that has been built according to stringent fire and construction codes. This is the message and the purpose of this book. Writing this kind of software is also expensive, but realtime development shops tend to follow rigorously defined methods to reduce risks and eliminate extraneous costs.

The author dives right into the specifics of operating-system design by chapters 3 and 4, with discussions about sharing resources, critical sections, and a very concentrated look at scheduling. Chapter 5 deals with reliability, a subject of much controversy in the microcomputer world. With realtime systems, operational reliability is so important that the design is centered on it. Automatic recovery facilities and watchdog processes are covered.

No one development methodology is a panacea, but neither is using several in a haphazard manner or using none at all. In chapter 6, Allworth mentions the more renown design methods and points out realtime concerns in a sample existing system. All of this in fewer than 10 pages. This treatment is completely inadequate for students of software or electrical engineering.

Allworth discusses predicting and measuring the performance of a system in chapter 8. Topics include performance factors, queuing theory, hardware monitors, and profilers. He says it is easy to write a colossal software system that runs too slowly for its application, particularly with realtime applications, so models should be considered before the system is built.

With regard to PC/AT software development, Allworth's text becomes a source of inspiration in dealing with environments like MS-DOS 4.0 from Microsoft, DRI's Concurrent PC-DOS or GEM, Microsoft Windows, and IBM's Topview. A user not working with one of these packages, however, need not bother with 80 percent of the book. These packages variously provide multitasking and support a number of terminals, windows, and/or graphics. The windows/graphics support will increase source code size about 20 percent. Multitasking enables developers to write code that is message-based rather than call-based for the applications that are better adapted to such design. The book contains information that probably can be applied to most software projects written for MS-DOS versions after 3.x, running on a PC with an 80286.

Although brief in its format, this book is nevertheless a very worthwhile introductory text for students of realtime software design.

—MICHAEL B. BENTLEY

TECH BOOK

ACCESSORIES/SUPPLIES

RESET BUTTON FOR IBM PC
The button IBM forgot. A good reset button is a basic tool for PC (PC/XT) programmers. When your machine is hung up, a touch of the button will quickly reboot your computer. This kit installs easily and mounts out of the way behind your machine. Easy instructions included. Send $19.80 (add 6% in MN).
RESET CORPORATION
P.O. Box 14809
Minneapolis, Minnesota 55414
(612) 489-5892

BUS. OPPORTUNITIES

EXTRA INCOME
.. Learn 3 steps to Profitable Consulting
.... How to Locate Prospects
.... How to Bid on Projects
.... How to Reference Sell New Clients
.. Discover 4 Profitable Client Services
Invest $9.95 in "Make Your PC Profitable."
FREE Table of Contents sent on request.
TULSA COMPUTER CONSORTIUM (TCC)
P.O. Box 707
Owasso, OK 74055
918-747-0151

Personal Computer Owners
CAN EARN $1,000 TO $5,000
monthly selling simple services performed by their computer. Work at home-in spare time. Get free list of 100 best services to offer. Write:
C.I.L.B.Q.
PO Box 60369
San Diego, CA 92106-8369

HARDWARE/ADD-ON BOARDS

DT2801 SERIES ANALOG I/O
Plug-in data acquisition boards with 8DI/16SE analog inputs, high or low level programmable gains, 2 analog outputs, 16 lines of digital I/O, DMA, on-board clock, and on-board microprocessor. Optional software subroutines and screw terminal panels.
DATA TRANSLATION INC.
100 Locke Drive
Marlboro, MA 01752
(617) 481-3700

$200 PER MEGABYTE
Memory expansion for your IBM PC, XT, OR AT at $200 per megabyte. All the memory you've dreamed of at a price you can live with. JRAM-2 with JRAM-AT hold up to 2 megabytes per board with Ser/Par/Clks also available. Megamemory and I/O ports all in one slot!

TALL TREE SYSTEMS

Tall Tree Systems
1120 San Antonio Road
Palo Alto, CA 94303
(415) 964-1980

FIXED DISK BIOS/BOOT
FiXT boots from most popular Hard Disks—DAVONG, TECMAR, IOMEGA, GT LAKES, etc. Adds XT-like BIOS interface to your disk for PC. Security, multiple volumes, removable media support optional. No-slot plug-in installation. Specify controller and computer with order. $80-$95. Add $3 shpg, CA tax.

GOLDEN BOW SYSTEMS
P.O. BOX 3039
SAN DIEGO, CA 92103
(619) 298-9349

SPEECH SYNTHESIS
SynPhonix: TRUE Unlimited Speech Synthesizer for IBM-PC/XT/AT/jr & compatibles. This low power short card includes an SSi263 speech chip, amplifier and speaker. Software includes Text-to-Speech, Phonetic Editor, Talking Clock & demos. Can be programmed with BASIC and other languages. Prices start below $200.

SynPhonix™
Electronic Speech Articulator

Artic Technologies
2234 Star Court
Auburn Heights, Michigan 48057
(313) 852-8344

HARDWARE/PERIPHERALS

A/D TO RS232 CONVERTER
Low cost stand-alone data acquisition board for any 8 bit RS 232 port. 8 channels, 8 bit conversion. 0-5V input, 300-9600 baud. Small size, 4" × 5", with wall plug power supply. Easy-to-use, includes sample driven software listings. Kit w/all parts & PS, $59.95 + $4 shipping. Assembled & tested is $89.95 + $4.
TPS SYSTEMS
14820 Elmore Road
Anchorage, AK 99516
(907) 345-6730

33% FASTER AT—$19.95
If you have a PC AT you can plug in a new crystal, and watch it run 1/3 faster. Non-technical 10 minute installation (no soldering), and no expansion slot usage. Compatible with all software. Satisfaction guaranteed. Check or Money order. COD. $3 extra. VISA/MC
AVTEK ENGINEERING
P.O. Box 21-171
Eagan, MN 55121
(612) 452-7415

HARDWARE/DISK DRIVES

1.2MB PC-AT DISK DRIVES
AT your PC by replacing your diskette controller and adding a high-density disk drive. JDISKETTE handles 1 to 4 diskette drives plus and I/O module. $249 for JDISKETTE-SA, $250 for TEAC 55GF (1.2MB); $200 for TEAC 55B (36-K); $180 for Ser/Par/Clk module.

TALL TREE SYSTEMS

TALL TREE SYSTEMS
1120 San Antonio Rd
Palo Alto, CA 94303
(415) 964-1980

SOFTWARE/ARTIFICIAL INTELLIGENCE

A.D.A. A.I. PROLOG-LISP TOOLKIT
A.I. Toolkit. A.D.A. PROLOG and Cybermetrics. UNX LISP in one package. Virtual memory PROLOG has full C&M syntax. LISP has macros, catch, throw, optional and rest parameters, structure editor and utilities w/source. Both systems have tree structured object list. Use with any text editor. $149.95
A.D.A.
1570 Arran Way
Dresher, PA 19025
215-646-4894

SOFTWARE/BUSINESS OPPORTUNITIES

RETRACTION
Below is the CORRECTED version of an advertisement which appeared in our May June July issues

SOFTWARE AUTHORS
WE WILL CONNECT YOU TO MAJOR SOFTWARE PUBLISHERS FREE. We are in contact with many publishers looking for marketable software. If you have a quality software package or program and wish to have it published and distributed, contact:
SOFTSEARCH, Inc.
P.O. Box 281
Budd Lake, NJ 07828
(201) 627-1790

VERTICAL MARKET OPPORTUNITY
If you develop software for the Advertising, Publishing or Broadcast industries call us. We're Interactive Market Systems. We have 10,000 clients in those industries World-Wide. If your product can benefit our client base, we'll provide marketing and sales support. If you currently have a product or are about to develop one give us a call. Contact Tami Peterson.
IMS Micros
355 Lexington Avenue
New York, N.Y. 10017
212-682-9494 1-800-223-7942

SOFTWARE/COMMUNICATIONS

FoReM PC Bulletin Board System
Advanced Bulletin Board System for IBM PC. Multiple message bases; Electronic Mail; Xmodem; 255 access levels; Remote sysop access; Privilege system; Hard disk support; Menu driven text file Data Base ... Much more!
Commnet Systems
7348 Green Oak Terrace
Lanham, MD 20706
(301) 552-2517

SOFTWARE/COMPILER

DeSmet-C - $109
Full K&R C compiler, linker, librarian, fullscreen editor, profiler, overlays, example software, 140 page manual. BOTH 8087 and floating point libraries. OUTSTANDING PRICE/PERFORMANCE. Consistently rated 1st or 2nd in published benchmarks. PC/MSDOS. No royalties on generated code. Newsletter. Updates $20 ea. $109 shipping included. Symbolic (C source level). Debugger Option $50 extra. C Compiler for the Macintosh $150.
C WARE CORPORATION
Dept. PCTJ
P.O. Box C
Sunnyvale, CA 94087
(408) 720-9696

RATES AND INFORMATION

Standard listings consist of a bold lead line (23 characters maximum); 7 lines of body copy (45 characters per line); 4 lines for company name, address and telephone number. $110 per insertion—3 issue minimum. Additional charge for extra lines and company logos. Prepayment and frequency discounts available. American Express, MasterCard, Visa accepted. Copy subject to publishers approval. Send typewritten or printed copy, reproducible logo art (if applicable) and remittance to Kathryn Cumberlander, Classified Sales Manager, Ziff-Davis Publishing Company, One Park Avenue, New York, NY 10016. For additional information, assistance, or to place an order by phone, call collect (212) 503-5115.

TECH BOOK

C LIBRARY + WINDOWS + BTREE

******* BEST YOU CAN GET ********
**** C POWER FUNCTION LIBRARY ****
Over 325 richly commented tested functions
ALL SOURCE CODE! .... NO ROYALTIES!
**** 4 DISKS: 400 PAGE MANUAL *****
51 screen handling/graphic funct; 50 cursor/keyboard/data input; 85 superior string functions; 28 system status & control; 70 utility/DOS/BIOS/time/date; 42 printer control.
No matter what ELSE you have, get this one
Lattice, Microsoft, CI86, Aztec any full K&R
COMPLETE LIBRARY $129.95

******* C POWER WINDOWS ********
PROFESSIONAL WINDOW MANAGEMENT
overlays, borders, popup menus/help windows
file-to-window & keyboard-to-window support
monochrome or color, three kinds of menus
status-line, color, highlighting, much more
C WINDOWS: Complete source code... $129.95

******* SUPERFONTS FOR C ********
Extra size dramatic on-screen character fonts Monochrome or color! Use our fonts or create your own ... Functions + font libraries .. $49.95
** B-TREE LIBRARY and ISAM DRIVER ***
Fast easy indexing! 16 million each: keys and records; variable length records! Multiple indices; examples; Full source code! PC-DOS, UNIX, XENIX, CP/M. Library only $79.95
ISAM ... $49.95, BOTH for only $109.95

Make utility: save time avoid errors $59.95

C WINDOWS + C-LIBRARY all for $179.95
add SUPER FONTS FOR C only ... $199.95
add B-TREE & ISAM ($440 value) ... $299.95

Entelekon
SOFTWARE SYSTEMS

Entelekon
12118 Kimberley
Houston, TX 77024
(713) 468-4412

SOFTWARE/DATA ENTRY

Data Entry System

Heads down data entry for the PC will allow fast and accurate preparation of data files for your mainframe. Features: Auto dup, verification, auto format change, table look-ups, range checks, batch totals, dup key, record insert, and much more. Ask us about our free 30 day trial period. Only $395.
COMPUTER KEYES
6519 193 SW
Lynnwood, WA 98036
(206) 776-6443

Semi-Custom® DATA ENTRY

Users frustrated by manuals, codes or forms? General purpose data entry software doesn't suit you? Add a front-end tailored to your specifications. Fully supported turn-key data entry module designed for occasional users. Fixed price. $1,000-$7,000. Users manual on a Business card! For free quote & sample, call:
SEMI-CUSTOM® SOFTWARE
6807 Brennon Lane, Suite 110
Chevy Chase, MD 20815
(301) 654-8135

SOFTWARE/ DEVELOPMENT TOOLS

THE FORMS DESIGNER™

Attention IBM Pascal, FORTRAN users! Save time in designing formatted screen I/O. Interactive Forms Editor allows you to draw lines and boxes, define fields, and edit text. Access forms or read keyboard entry by writing only one line of code. Provides sequential data retrieval and storage. Requires 128K RAM. Only $275 complete. Demo and manual $35. Call or write:
BIT SOFTWARE
PO Box 619
Milpitas, CA 95035
(408) 262-1054

TURBO-PLUS™

Turbo-Plus™ is a set of external procedures crafted in assembler language and designed specifically for Turbo Pascal's interactive environment. Turbo-Plus™ includes: File Handle Disk Access Method, Instant Memory Mapped Text Display, Expanded Text Display. $34.95 plus $5 s&h.
NOSTRADAMUS
5320 South 900 East
Salt Lake City, UT 84121
(801) 261-0769

COLOR, CURSOR CONTROL

DATE/TIME, KEYBOARD, PEEK, POKE, COMMAND LINE ACCESS, and more available for IBM/MS FORTRAN, IBM/MS PASCAL, and GENERIC (MS) BASIC on PC/DOS and many MS/DOS machines. No redistribution fees. $47 U.S. MN residents add 6% sales tax. BARTSUBS by BARTSOFT
3210 Lake Shore Boulevard
Wayzata, MN 55391
(612) 473-4032

REVISION CONTROL SYSTEM

The Software Revision Management System™ stores all versions of source/documentation in a single ASCII file. Allows retrieval of any version of source and application of changes, while recording when, why and where changes were made with no duplication of common code. MS/PC-DOS 2.0 $69.95 + $5 s/h. (MN + 6%.) MC/Visa.
QUILT™ COMPUTING
7048 Stratford Rd.
Woodbury, Minnesota 55125
(612) 739-4650

BASIC TOOLS

Screen Development Aid $29
- Create screens quickly (40/80)
- TEST screens prior to coding
- Automate screen functions from BASIC

SORTSUB—Sorts any array $19
- Sort a 1000 element array in 3 secs.

Interp. & Compiled BASIC, DOS 1.1-3.0 IBM PC, PCjr, XT, AT and compatibles
DEPENDABLE SYSTEMS
P.O. Box 191090
San Diego, CA 92119
(619) 287-3391

SoftRoms™

High Performance Graphics Tools. FAST; 200K pixels/sec avg. TIGHT; 14K. COMPLETE; 2/3D-Poly Lines, Bit Blocks, Windows, Fill, Arc, Icon, Character, String, Macros over 100 functions. Plus; assembler hooks, multiple language & hi-res boards, icon editor, memory draw & print, mouse & keyboard touch, type faces. Basic kit $70.00.
AFFINITY, INC.
126 Normal Ave.
Buffalo, New York 14213
(716) 882-5077

CROSSREF/BASIC & PASCAL

Debug programs, customise listings, control evasive variables! Our utilities will list your source, cross-reference variables in sorted order, and list procedure/function names. Available for Microsoft® Basic and Turbo Pascal® at $29.95 each or receive both for $49.95. All orders add $4.00 S&H. Check or Money Orders (Inquiries invited) to:
EXECUTIL
PO Box ### (12516)
Albany, N.Y. 12212

C TOOLS—MAKE

MAKE automatically keeps track of the dependencies between modules and include files. When you edit a file, you need only invoke MAKE to make sure that all the necessary modules are correctly compiled and linked and that object libraries are updated. Includes other useful tools. $40.00 check or COD.
INFOSOFT, Inc.
9020 Perryvale Rd.
Balto., MD 21236
(301) 256-2631

** PASCAL'S FRIEND **

PASCAL'S FRIEND v2 contains source code for use with IBM PC Turbo Pascal: 1-2-3 style menu routines, keyboard handling, save & restore screens, read disk directory or any track & sector, system clock and calendar routines, write strings in any attribute, DOS function calls, MORE! $49.95 check, MO, Visa, MC
J.S. COMPUTING
815 N. 12th St. Suite 5
Allentown, PA 18102
(215) 821-9020

SOURCE DEBUGGER FOR LATTICE C

The MSD DeBugger™ is the last, and perhaps final, word in programming assistance for Lattice C users. C Debugger produces a high-level view of C programs via function names line numbers, variable names and C data types, plus a low-level view of machine addresses and instructions for testing assembler language functions. More features include:
- All documentation is prepared for programmers.
- Online help screen throughout the process.
- Capability to single step through your program.
- Set break points, examine registers and variables.

$165.00 + $3.50 shipping VISA/MC

Cdebugger
a high level language debugger

MSD
214 1/2 West Main St.
St. Charles, IL 60174
(312) 377-5151

BTREES IN C WITH SOURCE

The Softfocus B-Tree Library is a complete set of data management functions written in K&R C. We support random and sequential access, multiple indices per data file and up to 16.7 million keys per index. Also, our software is customizable because we supply full source code. Manual & examples included.

B-Trees
$77.00

SOFTFOCUS
1277 Pallatine Drive
Oakville, ONT. CANADA L6H 1Z1
(416) 844-2610

THE HAMMER: TOOLS IN C.

More than just BIOS/DOS access, THE HAMMER also provides screen attribute control, 123-like menus, easy data entry & verification of dates, numbers, & strings, AND MORE. Super data entry routines work together in single or multi-field (FULL-SCREEN EDITING) mode. This is NOT just another general purpose library. Supports: CI-C86, Lattice, De Smet, MWC, and new Microsoft V3 C compilers. $195 wtih source & manual. Call or Write:
OES Systems
1906 Brushcliff Rd.
Pittsburgh, PA 15221
412-243-7365

TEST FOR HARDWARE BUG

If your IBM PC is less than two years old, chances are it has a SUPERBUG in its hardware that will let you do something that should be impossible —to BEAT THE COMPUTER AT AMERICAN ROULETTE. To test your PC for this superbug, order a copy of the ROULETTE TEST PROGRAM by sending a money-order for $5.00 (Canadian) to:
Hardware Debugging Services (1985),
135 Cedargrove Court S.W.,
Calgary, Alberta, Canada. T2W 4T6.

TOOLS FOR TURBO PASCAL

40+ DEVELOPMENT SUBROUTINES
Including Debugging and tuning Aids

SNAP NEX DUMP	SORT
SET FKEY CMNDS	DATE/TIME/DOW
ELAPSED TIMER	SECTOR 1/0

PC DOS 2.0 TURBO PASCAL 2.0 Source incl.
Send $29.95 check/MC/VISA (TX pay $31.78).
ALLEN, EMERSON & FRANKLIN
PO Box 928
Katy, TX 77449
(713) 391-8570

TURBO PROGRAM GENERATOR

PRODUCES READY-TO-COMPILE
TURBO PASCAL SOURCE
Be 10 times more productive—Pays for itself.
SIMPLY layout the screen & define the fields.
- Full-screen Entry; Automatic Field Edits
- Built-in Data Base Update & Retrieval
- Paint-the-Screen Editor, Color support
- Arithmetic, HELP, Tables, & more

User Manual & source included. Only $99.95.
Visa/MC.
ALLEN, EMERSON & FRANKLIN
PO Box 928
Katy, TX 77449
(713) 391-8570

TECH BOOK

C-INDEX™
C-INDEX is a state-of-the-art data management library for C. Ideal for data and text applciations. B+Tree, variable length records, multiple keys. We accept VISA, MC, Amer. Express cards. Money-Back Guarantee. Call or write for our free info packet.

- C-INDEX/FILE $99 Object Code
- C-INDEX/PRO $199 No Royalties
- C-INDEX/PLUS $395 Full Source

TRIO SYSTEMS
2210 Wilshire Blvd. Suite 289
Santa Monica, CA 90403
213/394-0796

PASCAL-TO-C-TRANSLATOR
Convert UC5D Pascal, MT+Pascal, and others to K&R C. Handles nested procedures, intrinsic functions, separately compiled units and modules, all data types including long integers. Requires 256K IBM PC/XT/AT. Send us up to 500 lines of Pascal and we will convert it for you FREE. Licensed source and executable code $5,000, executable only $2,500, or conversions 25 cents/line.

TGL
INC

TGL INC.
4400 Sulphur Springs Road
Corvallis, OR 97330
(503) 745-7476

FREE COBOL SOURCE CODE
Includes: on line 30 screen NOTEBOOK, Menu Driven Reviews IBM-PC COBOL. 13 SECTIONS to select from. A FREE COBOL Demo programs. OVER 1,800 lines of SOURCE. Covering: Inter-program comms, Graphics, Function Key and Special Register Usage: ERASE Disk files thru COBOL. ALL on diskette. $24.95 Much More.
Sunus Systems, Inc.
4244 Ridge Lea Rd Suite 25
Amherst, NY 14226
(716) 834-2820

CONTEXT-SENSITIVE HELP
TRUE/Helper adds context-sensitive help to existing programs. No access to program source is necessary. Works as a pop-up under the target program. Help text can be built using any editor. Text is compressed to reduce space. Works with most custom and off-the-shelf software. $39.95
GRW Software
P.O. Box 440007
Houston, Texas 77244
(713) 556-9878

C-LIBRARY FOR TI-PRO
For Lattice C Programmers
—ROM BIOS Full crt/keyboard/disk/printer
—DOS Functions
—Graphics functions written in assembler
—Some functions are BASIC equivalent
—Outstanding documentation w/examples
—Over 100 useful functions $99
Programmer's Components
335 E Penna Ave
Southern Pines, N.C. 28387
Phone (919) 692-2974

ROMable CODE on PC!
PCLOCATE allows PC users to develop ROM-based software from MS-DOS. "Exe" files. The user specifies the physical location of all segments. Output files are compatible with most PROM programmers. PCLOCATE supports the 8086, 8088, 80186, 80188, and 80286 processors. MC/VISA.
ALDIA SYSTEMS INCORPORATED
P.O. Box 37634
Phoenix, AZ 85069
(602) 866-1786

SOURCE CODE LIBRARY
.."Structuring BASIC Programs" comes with:
....BASIC examples on DOS 2.0. Diskette.
......BASIC Check Writing Program.
........BASIC Source Code Library
..........HOW to Structure Package only $9.95.
FREE Fact Sheets.
FREE Contract Programmer Registration.
TULSA COMPUTER CONSORTIUM (TCC)
P.O. Box 707
Owasso, OK 74055
(918) 747-0151

MATIS AT ONLY $49.95
Create large virtual screens; formatted data entry field; windows for single or multiple screen display. Total control of video attributes. Resident DOS extension complete with interfaces to BASIC, C, PASCAL, and Assembler. Credit card orders: 800-227-2400 ext. 989. Add $5 s/h. Also MATIS/T for Turbo Pascal $29.95.
Softway, Inc.
500 Sutter St. Suite 222T
San Francisco, CA 94102
(415) 397-4666

Announcing SOURCE-IT™
Increase your BASIC programming productivity. SOURCE-IT is an advanced BASIC cross reference utility that provides a printed listing of your source code with all line references listed next to the referenced line. All variables, files, calls and Compiler Metacommands are also listed. It's invaluable! $89.99.
SOFTWARE INNOVATIONS, INC.
16621 Dolores St.
Huntington Beach, CA 92649
(714) 840-6411

TURBO PROFESSIONAL™
Turbo Pascal Tools
*SERVICE INTERRUPTS IN PASCAL
*MAKE YOUR ROUTINES RESIDENT
*VARIABLE SIZE WINDOWS
*KEYBOARD MACROS
*DOS PROGRAM EXECUTION & RETURN
*SOURCE CODE INCLUDED
*11 INCLUDE FILES
*109 ROUTINES
*SUPER MACS EXAMPLE PROGRAM
Super Macs features: define macros, save macros, load macros, movable window, prints concurrently in DOS 3. Works within other programs. Available separately w/o source code.

SUPER MACS ONLY (.COM)	—$10.00
WITH TURBO PROFESSIONAL & SUPER MACS SOURCE CODE	—$49.95
SHIPPING & HANDLING	—$ 5.00

SUNNY HILL SOFTWARE
13732 Midvale North, Suite 206
Seattle, WA 98133
(206) 367-0650
Requires 100% compatible, DOS 2-3, Turbo Pascal 2-3 for compatibles.

TURBO EXECUTIVE™
For Real-Time Multi-tasking, control applications. Interrupt driven, Turbo Pascal source code includes a background ASYNC COM task, memory resident code example, windowing, video and directory utilities!

- Non Pre-emptive version $64.95
- Pre-emptive (time slice) version $164.95
- Call or write for FREE brochure consulting and programming service available.

Clear Creek Computing
2107 Audubon
League City, TX 77573
(713) 332-9940

SOFTWARE/EDIT

PQfEDIT
Full-screen editor, EDLIN alternative for the PC family. With: 4-way cursor, insert/delete/replace char, 4-way scroll, split/join lines, add/copy/move/delete lines, search/replace strings, get/put to disk, help screens. Edit multiple files at same time. Needs 128K and MS-DOS. $45 (+$1.80 tax in VA) + $2 shipping.
ADAM Systems Corporation
5919 Munson Court, Dept. 404
Falls Church, VA 22041
(703) 379-0669

LIL' JAKE
So long EDLIN! Hello Lil' Jake!
A Command-Free Full Screen Editor for IBM PCs and compatibles. Comprehensive on-line help facility; global searches/changes, file sorts, split/join; editing several files simultaneously; DOS commands within editor; user customization; and more. $49.95 + S&H.
Fooks Mathewson Company
P.O. Box 2424 (Oper-54)
Arlington, Virginia 22202
(703) 684-8284

SOFTWARE/ ENGINEERING

smARTWORK®
Printed-circuit artwork editor for double-sided boards up to 10" by 16", runs on an IBM-PC. Color display allows complete interactive control over the placement and routing process. 2X artwork can be made on a dot-matrix printer or pen-and-ink plotter, $895. Write or call for a brochure.

WINTEK

WINTEK Corporation
1801 South Street
Lafayette, IN 47904-2993
(317) 742-8428

METAL FABRICATORS
PC/Cutlist takes input from your bill of material—Detail drawing and calculates the best cutting combination for any length stock and prints a shop ready cutting list and scrap report. Also an optimization feature finds best mult length for mill orders. Price $300. Demo Disk $25.00
THE JOSEPH ALBERT CO.
P.O. Box 611
Blue Island, Illinois 60406
(312) 349-9032

KALMAN FILTER PRIMER
Learn Kalman Filter with this software, document, and support package. Easy-to-use programs provide practical hands on experience. Unique step-by-step approach designed for self-study. Many numerical examples with straightforward explanations. For engineers, scientists, programmers, and managers.
KALMAN FILTER PRIMER
22338 Lull Street
Canoga Park, CA 91304
(818) 716-0816

STEAM
STEAM computes the thermodynamic properties of water over a wide range of values commonly used in applied thermodynamics. The FORTRAN package includes an interactive "stand-alone" program and code in object form, callable as a subroutine. Specify your FORTRAN compiler. $295. (Calif. + Tax).
SOFTset, INC.
10889 Wilshire Boulevard, Suite 810
Los Angeles, CA 90024
(213) 824-1904

CREATE PCB's EASILY
Easily create double-sided printed circuit board artwork of up to 21" square using PC-layout on an IBM-PC. The artwork is created using a color artwork editor. the final artwork, solder mask and silk-screen (text) is produced using either a printer or a plotter. Price $745. Demo disk/manual $95.
Draco Technology
7210 Jordan Ave., Suite D50
Canoga Park, CA 91303
(800) 235-6646 ext. 646 (outside CA)
(800) 235-6647 ext. 646 (CA only)

SOFTWARE/GENERAL

C Users' Group

PUBLIC DOMAIN SOFTWARE IN C
Over 45 volumes of public domain software in CP/M & MS-DOS formats.

- editors & compilers
- text formatters
- communications packages
- many UNIX-like tools

Write or call for more details.
THE C USER'S GROUP
415 E. Euclid Box 97
McPherson, KS 67460
(316) 241-1065

EXCITING SOFTWARE GIFTS!
After a successful 1984 season GREETING-WARE® IS BACK with sing-along music plus software goodies. Ideal for Christmas and Birthdays. We have an exciting line of attractively priced software gifts for your favorite IBM PC user. For our FREE catalog, write to
ROXBURY RESEARCH INC.
RD 1, Box 171A, Roxbury, NY 12474
or call NYS: 607-326-4070
Outside NYS, call toll free 800-DIALROX

TECH BOOK

SOFTWARE/STATISTICS

RATS!

RATS is the ultimate econometric software package. It is powerful, fast, accurate, and inexpensive. RATS performs regression analysis, time-series forecasting (including Box-Jenkins) and cross-section (including logit and probit). RATS supports a variety of data formats, including Lotus WKS files. RATS also offers a programming language that provides maximum flexibility and power. Spectral analysis and graphics are available as options. Base program $200. Demo including 300 page manual $40. MC/VISA.
VAR Econometrics
P.O. Box 19334
Minneapolis, MN 55419
(612) 822-9690

SOFTWARE/STOCK ANALYSIS

STOCK OPTION TOOL KIT

Use Monte Carlo analysis and random walk theory on stock option strategies. The kit has 50 pages of documentation, bibliography and 11 programs to evaluate covered calls, naked puts and calls, and various spreads. Price $199. Demo disk with one program $25. IBM PC, PC XT, 128K, DOS 2.0+ check or MO.
EXATECH SOFTWEAR
939 Pulpit Rock Circle North
Colorado Springs, CO 80907
(303) 599-0280

PC-STOCK GRAPHIC ANALYSIS

Graphic trend analysis aiding in technical stock, bond, and commodity analysis. Includes "ON BALANCE VOLUME" and moving average. Ideal for the small investor. PC-STOCK comes complete with source code and examples. Requires IBM-PC, 128k, DOS 1.1-2.1 and hi res graphics. $30 + $3 shipping.
SUNRISE ENTERPRISES
P.O. Box 2610
Goleta, CA 93118
(805) 968-4170

SOFTWARE/TYPESETTING

HIGH-TECH TYPESETTING

Transmit your text via toll-free lines directly to our fully automated typesetting system. $2. per K characters with a $5. minimum. Same day service. 200 typefaces in sizes up to 72 point. Send $15. + $3. shipping for our 220 page guidebook, or call toll free and use your MC, VISA or AMEX.
INTERGRAPHICS INC.
106A South Columbus Street
Alexandria, VA 22314
(800) 368-3342 or (703) 683-9414 in DC area.

SOFTWARE/UTILITIES

DISK MECHANIC

THE ULTIMATE Floppy Disk Backup & Repair Utility. Can back up ALL software protected disks written on the IBM PC. Works manually or automatically. Files or sectors can be restored, searched, examined & changed. Checks disk drive speed. req. IBM PC, DOS 1.1 (128K)/2 (192K) + 64K if only 1 floppy drive. $73 ppd. USA
MLI MICROSYSTEMS
PO BOX 825, Dept. TB
Framingham, MA 01701 USA
(617) 926-2055 for info MC/VISA

NOBLINK™

ErGoNoMiCs! Eye strain? With just a keystroke you can change your blinking cursor into a friendly solid block reverse video cursor. If you wish you can change it back and forth (Blink/No blink) right in the middle of a program, such as your word processor. Give your eyes a rest. $29.95 plus $5 s&h.
NOSTRADAMUS
5320 South 900 East
Salt Lake City, UT 84121
(801) 261-0769

Hard Disk DIRECT ACCESS™

The ULTIMATE utility for any hard disk user. DIRECT ACCESS is a powerful hard disk management tool. Organizes your software programs into a "user defined" menu system. Features attractive screen display, rapid access into an application using a single keystroke & easy to learn. Also handles batch files & complicated DOS commands. Order today. Only $59.95 + $5.00 shipping & handling. MC/VISA.

DELTA TECHNOLOGY

DELTA TECHNOLOGY
P.O. Box 1104
Eau Claire, Wisconsin 54702
(715) 832-0958

AVOID FRUSTRATION!

PC-CATLG—easy to use, reliable, Diskette Management Utility. Directory info for EACH file is read/stored in a master catalog with a 40 char user comment. Functions: ADD/CHG/DEL/SRCH/UPD/DISPLAY/PRT/LABELS. Chg VOL labels. Req. DOS 1.1/2.X, 256K, 80 mono/color, 2 DSDD drvs (SSDD add $3), parallel printer opt. $25.
Oregon-microTech
1717 Magnolia Ave.
Medford, OR 97501
VISA/MC (503) 773-8327

COMPUTER SYSTEM LOG

Monitor activities on your computer with SYSLOG. The System Log Utility is a command driven nonresident DOS extension that allows automatic task recording. With SYSLOG, all of your computer time is noted, saved, and categorized. Reports printed for tax purposes or for monitoring PC usage. Only $34.95 + Sh. VISA/MC accepted.

SYSLOG
SYSTEM LOG UTILITY

Apex Resource, Inc.
23 Christine Court
Stormville, NY 12582-9612
(914) 221-2611

UNIVERSAL CROSS REFERENCER

One Cross Referencer does it all for BASIC, C, Pascal, FORTRAN, Modula2, COBOL, DBASE-II, Macro Assembler, etc. User-defined page titles, page numbers, file date/time. Unlimited symbol length, adjustable output margins, compressed print. Multiple input files, Full directory and path support. DOS 2.x, IBM PC/PC-XT, 128k, $39.95.
DALSOFT Engineering
3565 High Vista
Dallas, Texas 75234

DOS PATH COMMAND FOR DATA

Now Lotus, d-Base, WordStar etc. can access data files no matter where they're located on your hard disk, floppies or ram disk. DPATH lets you organize your directories conveniently for you, not just your programs. Requires only 4K. Works on MS-DOS 2.0, 3.0 $25 + $2 s/h Visa, MC, company PO, check.
PERSONAL BUSINESS SOLUTIONS INC.
P.O. BOX 757 DEPT. T
Frederick, MD 21701
301-865-3376

AT/XT/PC HARD DISK EXPANSION

Replace hard disk with a bigger one, or add a second drive! Vfeature supports high capacity drives on standard AT, XT, and compatible hard disk controllers. Includes multiple volumes, security features, selectable clusters, keyboard lock. $80-$120 + $3 shipping + CA tax.

GOLDEN BOW SYSTEMS
P.O. BOX 3039
SAN DIEGO, CA 92103
(619) 298-9349

COMPRESS/ENCRYPT FILES

TRUE/Miser compresses/expands ASCII files to reduce storage requirements. Specifically designed for the compression of long text and source code files. Reduces file size by up to 80%. Optional data encryption protects valuable data from unauthorized access. No limit on file sizes. $19.95
GRW Software
P.O. Box 440007
Houston, Texas 77244
(713) 556-9878

DISPLAY ANY MS-DOS FILE

List ASCII and binary files. TRUE/View can be invoked as a pop-up or directly from DOS. Switch between ASCII and hex modes. View multiple files. Search for string patterns. Mark file positions for easy retrieval. Make small changes with edit feature. User configurable. $19.95
GRW Software
P.O. Box 440007
Houston, Texas 77244
(713) 556-9878

TRUE PATH UTILITY

Never misplace files again. TRUE/Path extends the MS-DOS PATH utility to search for data files as well as programs. Unlike other path programs, TRUE/Path uses the same path given by the MS-DOS PATH command to locate data files. Works with all MS-DOS software. A perfect for inexperienced users. $19.95
GRW Software
P.O. Box 440007
Houston, Texas 77244
(713) 556-9878

dBASE II CROSSREFERENCE

Crossreference dBASE II data fields and variables with the DBXREF command. DBXREF has full wildcard support. Output can be directed to the screen or printer. DBXREF will crossreference one, multiple or all memory variables and data fields in a program. DOS or CP/M, $27 + $3 shpg MC/VISA.
Edward Foster & Company
1 West Illinois Street
St. Charles IL 60174
(312) 584-2989

TURBO PRINT FILTER

Issue commands to EPSON/IBM Printers from Turbo Pascal's Editor via the DOS I/O pipe. Let Turbo's Editor be your WORD PROCESSOR. So easy—2 key strokes invoke a command!!
CONDENSED EMPHASIZED ITALICS
PAGE NUMBERS LINES PER INCH MARGIN
DOUBLE-WIDE DOUBLE STRIKE and more
$29.95 visa/mc Includes the source code
AEF
P.O. Box 928
Katy, TX 77449
(713) 391-8570

FKEY NOW AVAILABLE

The famous FKEY key processor is now running on PC, DOS/MS DOS. FKEY allows the redefinition of each key and key macros. No royalties on the run time package for product developers. Attractive dealer discounts and dealers are invited. Call or write for more information about FKEY. $39.95.
DITj
P.O. Box 60757
Sunnyvale, CA 94088
(408) 737-9000

TECH BOOK

SOFTWARE/GRAPHICS

TEKTRONIX 4010 EMULATION
High resolution screen/printer graphics for the IBM PC. Full interactive capabilities with file transfer and cross-hair control. Utilizes PLOT 10, protocols. OFF-line review of graphics output. Supports IBM color/graphic or Hercules cards. Easy to use 4010 emulation at an affordable price. $80.00. Demo disk $5.
Technological Systems Group
5044 Haley Court
Lilburn, GA 30247
(404) 923-4980

GRAPHICS LIB FOR TECMAR
TEK-MAR lets you do high-res graphics on your TECMAR Graphics Master. TEK-MAR is a library for use with MS Fortran. Features windowing, viewporting, clipping, axis rotation. Similar to Tektronix graphics. Includes screen dump/restore, Epson screen print, support for Hewlett Packard, Western Graphtec plotter. Requires MS-FORT 3.20, 320K, GMDEV.SYS (5.2). Price: $195.
ADVANCED SYSTEMS CONSULTANTS
18653 Ventura Boulevard, Suite 351
Tarzana, California 91356
(818) 407-1059

MetaWINDOW™
Advanced graphics toolkit provides Xerox Star/Apple MacIntosh style graphics on your IBM PC. Supports most popular graphics cards. Allows you to create pop-up menus, windows & icons; use proportionally spaced fonts; rubberband & drag lines, text or bitmap images; supports mouse/cursor tracking. Tightly optimized for use with Turbo Pascal, IBM Pascal, C, Fortran & Compiled Basic.
METAGRAPHICS SOFTWARE CORP.
444 Castro Street, Suite 400
Mountain View, CA 94041
(408) 438-5352—Telex: 4998586 METAGRAF

PLOTTER SOFTWARE
For the IBM and APPLE computers to drive the Hewlett-Packard and compatible plotters. The DGI TYPE SHOP will almost transform your plotter into a type setting system. The DGI ORGANIZATION makes it easy to display and update your organization chart. To order or for additional information contact:
DECISION GRAPHICS, INC.
P.O. Box 2776 PC
Littleton, CO 80161
(303) 796-0341

SOFTWARE/INTERPRETERS

C-Terp™-C Interpreter
A COMPLETE DEVELOPMENT ENVIRONMENT. Full K&R language and libraries; Complete built in screen editor;
compiler compatible, supports object library compiled with C86 or Lattice C or assembler; Multi-module support, includes global search; High-level symbolic debugging; Batchmode; Incredibly fast linking and semi-compilation; Runs on IBM-PC Dos 2.X, 3.X, 192K and up. Price: $300 (Demo plus manual $45) MC, VISA. See our display ad for more info.
Gimpel Software
3207 Hogarth Lane
Collegeville, Pa 19426
(215) 584-4261

SOFTWARE/LANGUAGES

IBM PC PASCAL/DOS 3.0/1
For the serious PASCAL and DOS user. EXAMPLES PLUS provide a rich set of examples and directly usable PASCAL function/proc code for PASCAL with DOS 2.0/1. EXAMPLES PLUS 2 provides additional info for PASCAL 2 with DOS 3.0. EXAMPLES PLUS 3 shows the differences and glitches in DOS 3.0/1, plus info on TopView, networking, extended graphic board and the AT hardware. Priced at $30 each, 2 for $48, 3 for $60. (CA add 6%).
JATO INC.
24861 Alicia Parkway, Suite 272-B
Laguna Hills, CA 92653

EASY WAY TO MASTER DOS!
New handbook shows how, why, and when to use all 44 PC & MS DOS commands quickly and easily! Lively hands-on tutorial makes learning fun, for a change. Covers basic thru advanced topics. All versions up to 3.1. Lots of practical examples. 221 pages only $15.95 at bookstores, or send check + $1.00 S&H to:
LITTLE, BROWN & COMPANY PUBLISHERS
200 West Street, Code T-368121
Waltham, MA 02254
1-800-343-9204 (credit cards)

ASSEMBLY PROGRAMMERS
Your library is here. Graphics, Floating Point Math, Trigonometry Functions, Windows, 8087 Support, plus much more. Supplied on IBM PC-DOS diskettes for only $49.95. Source code available for an additional $99.05. Written in MS Assembly Language for DOS 2.x or newer. Get yours today!!
BC Associates
13073 Springdale St., Suite 134
Westminister, CA 92683

SOFTWARE/OPERATING SYSTEMS

pcSHARE MULTI-USER O/S
pcSHARE allows your IBM-XT or compatible to support up to 4 users running 123, dBASE, WordStar, etc on inexpensive serial CRTs. For the software developer, pcSHARE efficiently runs compiled Basic, Pascal & C programs with full DOS 3.0 record locking. No risk 30 day money back guarantee.
DIGITROL COMPUTERS, INC.
440 Phillip Street
Waterloo, Ontario, Canada N2L 5R9
(519) 884-4541

CX/PC C EXECUTIVE
Real-time, priority task scheduler. Functions to queue inter-task messages and pointers. Example tasks. Interrupt-driven timer, serial and parallel port interfaces. Runable demonstration programs. DeSmet C source included; no license for generated programs. Described in June '84 IEEE Micro article. $75. Brochure.
INTR-SOFT CO.
P.O. Box 351
Bedford, MA 01730
(617) 369-6242

QUADDRIVE
Extends PC-DOS to support 80 track 96tpi drives for a total of 806K per disk. Mix and match 48 and 96tpi drives and optionally boot from a 96tpi drive. Reads/writes/formats 48tpi in a 96tpi drive. Step speed (down to 2msec) is set for each drive. Requires DOS 3.x. $U49.00 Free Air Post. CHECK/MC/VISA. Dealer inquiries invited.
Digital Business Systems
P.O. Box 33
RIVETT ACT 2611 AUSTRALIA
011-61-62886520

SOFTWARE/SECURITY

SECURE AT/XT/PC
Control system access, data access! FiXT/S. Control system boot for most popular XT/PC hard disk controllers. Vfeature for AT-and-XT-compatible HD controllers segments hard disk by volumes, controls access with passwords, supports hard disk expansion. $70-$130 + $3 shpg. plus CA tax.

Golden Bow Systems
P.O. Box 3039
San Diego, CA 92103
(619) 298-9349

SOFTWARE/SERVICES

TAPE TO DISK CONVERSIONS
Convert any 9 track magnetic tape to or from over 200 formats including 3½", 5¼", 8" disk formats and word processors. Formats available include IBM-PC, Apple, Altos, TRS 80 8" CP/M, Displayer., IBM Sys/??. Macintosh, Wang, Lanier, OS/6 and 300 more. Disk to Disk conversions also avail. Call for more info.
PIVAR COMPUTING SERVICES, INC.
47 W. Dundee Rd.
Wheeling, IL 60090
(312) 459-6010

DATA CONVERSION ON PREMISES
Keep your data confidential. TO/FROM cards, disks, tapes. ASCII/EBCDIC, 800/1600/3200BPI 9-Track. For IBM-PC/XT/AT and the compatibles. Hard disk and Bernoulli Box owners, we will bring our systems and do the conversion on your premises.
DP Modern Systems
804 North Reeder Avenue
Covina, CA 91724
(818) 966-2628

SOFTWARE/SCIENTIFIC

HP-PC HYPER-CALCULATOR
Activate emulation of an HP-11C programmable scientific calculator from any program, and return with the results. Features program and data storage on disk, indirect addressing, alpha-numeric messages, printing, on-line HELP, 100 registers, and 1,000-step programs for solving roots, numerical integration, complex numbers, matrices, base conversion, time/money problems. $39.95.
SUNDERLAND SOFTWARE ASSOCIATES
P.O. Box 7000-64
Redondo Beach, CA 90277
(213) 540-2567

SOFTWARE/SPREADSHEET

FORMAT—1-2-3
- **Loads ANY ASCII data into Lotus™!**
- 1-2-3 and Symphony Utility!
- Send spreadsheets using E-Mail!
- On-Line Help System!
- Runs on PC, XT, AT, and lookalikes!
- Also TI Professional, AT&T
- Extended Warranty Available!
- DOS 2.0 + with 64K, 1 drive
- $150, MC, VISA, CK, COD

RELIABLE SOFTWARE COMPANY
P.O. Box D
Titusville, FL 32781
(305) 267-2043
(800) 235-6646 Ext 581 (nat)

TECH BOOK

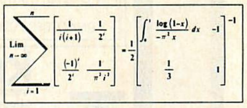
CONVERT ANY ASCII FILE
The DataMessager-the ESSENTIAL ASCII file conversion utility. Handles up to 256 simultaneous string search/replace operations in any file plus two levels of character translation. Supports wildcards and smart buffering, too. Menu driven-fun to use, easy to learn. Unlimited applications in wordprocessor/printer command conversion, file interchange, communications. PC/PCjr/XT/AT w/192k, 80 col. Only $155 postpaid.
R.A. Handverger
281 Village Street
Medway, MA 02503
(617) 533-2302

EASY AIM-30 day trial
Installs your programs on a hard disk and creates subdirectories, paths, and displays programs in a menu, all automatically. Features include time log, backup to floppy or tape, copy files to/from floppies to/from/between menu selected directories. DOS is transparent to end user. Powerful yet simple. $84.95.
LARSON SYSTEMS, INC.
BOX 193
STORY CITY, IOWA 50248
(515) 733-2717

IBM USES CPM DISKS!!
Uniform lets you use CPM disks in your IBM B: drive while running any PCDOS/MSDOS program. Read and write to the CPM disk directly or transfer files to IBM format. Over 70 formats also 8" and 96 TPI, IBM PC, XT, AT. Call for details. Dealers invited. (Not for Apples). $59.95 + $3 S&H. Uniform by Micro Solutions.
BLUE HERON
1108 South Second St.
Dekalb, IL 60115
(815) 758-2355

SOFTWARE/WORD PROCESSING

FORMATH™ TEXT-FORMATTER
For technical texts. Sizes & places equations, matrices, ratios, integrals, big symbols, footnotes. Macros, multiple fonts, 158 Greek/math symbols. Automatic hyphenation, section/equation/reference numbering, indexes, table of contents. Dot-matrix, daisywheel, laser printers, plain/graphics monitors. $400. $50 for demo.



SHANTHA SOFTWARE, INC.
50 West 97th St. Room 11N
New York City 10025
212-222-7647
Touch-tone Toll-free: 950-1088-wait-FORMATH

VOLKSWRITER DELUXE
Features TextMerge™ for customized mailing lists; multi-lingual, engineering & scientific symbols; horizontal scrolling to 250 characters; document size 1 megabyte; on-screen page endings, underline, boldface, double-strike, strike-through, multiple fonts & colors—fewest keystrokes per function. Only $295.
LIFETREE SOFTWARE INC.
411 Pacific Street
Monterey, CA 93940-2790

PC BUYERS GUIDE
THE INDEPENDENT GUIDE TO PRODUCTS AND SERVICES FOR IBM Personal Computers. This master directory lists, describes, rates for ease of use, and tells you about the latest in software, hardware, accessories, supplies, books, services, et al. Everything for you and your IBM PC. It even includes an Index of Computer Camps for both adults and children. From the publishers of PC. An ideal gift at only $7.95 (+ $1.55 shipping/handling) to U.S. destinations; and $11.00 elsewhere. Residents of CA, CO, CT, DC, FL, IL, MA, MD, NJ, and NY add applicable sales tax. Checks or MasterCard or VISA acceptable (with number, expiration date and signature). U.S. funds only. Mail to:
PC BUYERS GUIDE
CN 1914
Morristown, NJ 07960

TECH MART

FORTRAN or PASCAL PROGRAMMER?

READ THIS AD

QNX — NEXT DAY FREE DELIVERY *

QNX 1.2 Development System	595
QNX 2.0 Networking Version	1190
QNX Networking Board	425
SOFPLAN Multi-User Spreadsheet	200
ZIM Multi-User Database	1095

8087

8087-3 (5MHZ) Co-Processor Kit	159
8087-2 (8MHZ) Co-Processor Kit	315
80287 (AT) Co-Processor Kit	280
FORLIB: Fortran Scientific Subroutines	99

* QNX ORDERS OVER $500

kanatek MICRO
Moodie Drive High Tech Park
105-215 Stafford Rd. Ottawa, K2H 9C1
(613) 726-1907
Quantity discounts Dealers welcome

CIRCLE 398 ON READER SERVICE CARD

RST TOPVIEW SPOOLER

TV SPOOL is a TopView specific program which provides printer spooling capabilities normally found on large operating systems.

TV SPOOL incorporates the same user-friendly menu style as TopView and enhances the TopView environment with a flexible mechanism for printing files.

- Spool Queue Management
  - 100 Entries • Disk Resident
- Previous Page Buffer
  - Backup and Advance
- Banner Processor
  - DOS 3.1 Separator Format
- Flexible Options
  - Delete After • Multiple Copies

INTRODUCTORY PRICE IS $60.00
Visa, M/C Accepted

RIX Softworks, Inc.
17971-E Skypark Circle • Irvine, CA 92714

(714)261-0440

CIRCLE 391 ON READER SERVICE CARD

GRAPHICS
- Text/graphics generics
- 2D interactive
- 2D plots (full support) including auto-scaling, auto-axis generation, auto-labeling, tabular/log/parametric curves, splines, bars, pies, you name it, we have it.
- 3D Plots incl. 2 hidden line removal options—not just old-fashioned wire frame.

PEN PLOTS
- Standard plotter primitives plus
- FULL 2D support plus
- Interface to screen graphics,
- Limited 3D.

Clear and complete documentation.

GRAFMATIC	$135.
PLOTMATIC	135.

MICROCOMPATIBLES
301 Prelude Drive
Silver Spring, MD 20901
(301) 593-0683

CIRCLE 400 ON READER SERVICE CARD

PERFECT COPIES GUARANTEED

Why risk duplicating your important programs on your computer, when *our* equipment is designed solely to duplicate disks and verify their perfection 100%?

Over 600 formats. 3½", 5¼" and 8". Plus serialization, copy protection, labeling, packaging, shrink-wrapping and *fast,* personalized service.

Western Transdata Inc.
DISK DUPLICATION

1701 E. Edinger Ave.
Building A7
Santa Ana, California 92705
(714) 547-3383 (Collect)

CIRCLE 396 ON READER SERVICE CARD

Fantastic Price!

NASHUA DISKETTES

FIVE YEAR WARRANTY

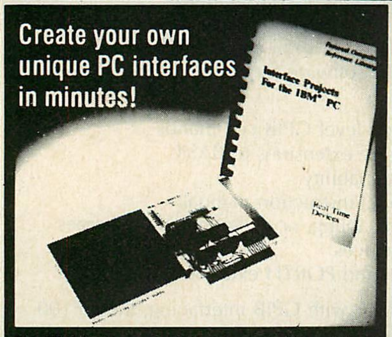
75¢ each Qty. 100

5¼" SS/DD or DS/DD, your choice, with hub ring and Tyvek sleeve, bulk packaged, no labels. Shipping extra. For quantity 50, add 10¢ each.

Get the same low price our high-volume duplication customers get!

CALL TOLL FREE
1-800-321-4668
in Colorado, 303-234-0871
ALF Products Inc. • Denver, CO

Create your own unique PC interfaces in minutes!



The PD100 allows rapid development of specialized PC interfaces. It features a buffered data bus, switch-selectable address decoder, and a large prototype area (up to 40 IC sockets). The 116-page manual covers basic interfacing concepts and details implementing A/D, D/A converters, I/O ports, connection of transducers and dozens of useful circuits.

PD100 w/manual - $99 plus $3.50 P&H	REAL TIME DEVICES
Manual only - $20 Postpaid	1930 Park Forest Ave. P.O. Box 906
PA residents add 6%	State College, PA 16804
Check, MasterCard, or VISA	(814) 234-8087

MAKE

make - a program/document builder. Make is used to correctly build an up-to-date program with a single DOS command. A program being up-to-date depends on each subprogram being up-to-date; each subprogram may depend on other subprograms being up-to-date. Make remembers when and how to update each subprogram through file creation dates, and user-supplied dependency lists and DOS commands. Similar to the UNIX™ make utility. Includes macro capabilities. Make can also be used to build up-to-date documentation.

This utility is part of a toolbox developed by DOS experts (one of whom has written articles in BYTE). It has been tested under DOS 2.x / 3.x (including DOS 3.1) on the IBM™ PC/XT/AT.

$49 for unprotected binary
(source is available)
$10 for manual only

For quickest response, order with Visa/MC by calling
800-621-2283 ext 912
In TX: 800-442-5321 ext 912

Or send check/money order to address below.
Add $5 shipping for binary, $3 for manual only
(non-USA orders, add $12 for binary, $10 for manual only).
TX orders add 6 1/8% sales tax.

PC Network Specialists, Inc.
Dept. T510, 13109 Chandler Drive
Dallas, TX 75243
214-373-2386

FAST SCREEN OUTPUT FOR TURBO PASCAL

FASTSCREEN ™
is a set of **inline** assembler and Pascal procedures for Turbo Pascal users.

- Display an entire screen or window almost instantly
- Process multi-field input screens that give your user full cursor control
- Sample program uses Conway's **LIFE** to illustrate use of procedures
- All source code included
- IBM PC/XT/AT and true compatibles

29⁹⁵ VA residents add 4% sales tax

TECHNISOFT
1710 Allied Street, Suite 37
Charlottesville, VA 22901
(804) 979-6464

Turbo Pascal is a registered trademark of Borland International

CIRCLE 397 ON READER SERVICE CARD

INDEX TO ADVERTISERS

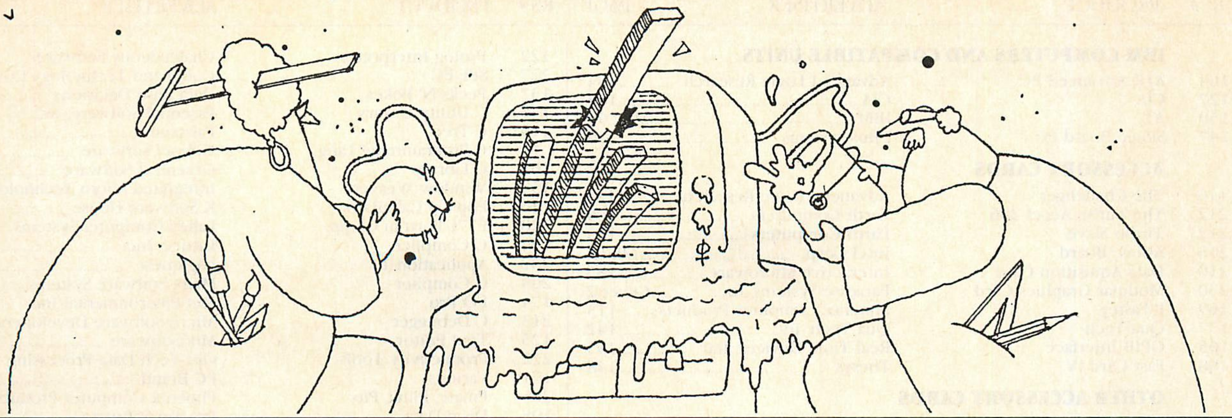


ILLUSTRATION • DAVID POVILAITIS

OCTOBER

October 2-4
First International Conference on Optical Storage Technology
Tokyo, Japan
Sponsor: TOC and USACO Corporation
Contact: Technology Opportunity Conference, P. O. Box 14817, San Francisco, CA 94114; 415/626-1133

October 3-5
UNIX Systems Expo 85—Fall
Boston, MA
Contact: David Small, Computer Faire, Inc., 181 Wells Avenue, Newton, MA 02159; 617/965-8350

October 5-8
5th Symposium on Small Computers in the Arts
Philadelphia, PA
Contact: Symposium, P. O. Box 1954, Philadelphia, PA 19105

October 7-9
10th Conference on Local Computer Networks
Minneapolis, MN
Contact: Dan E. Gahlon, Interactive Systems/3M, 225-4S-06 3M Center, St. Paul, MN 55144

October 7-11
Compsac 85
Chicago, IL
Contact: Miriam L. Holden, Compsac 85, Arqonne National Laboratory, 9700 S. Cass Avenue, Argonne, IL 60439; 312/972-5585

October 14-16
ACM 85: The Range of Computing, Mid-80s Perspective
Denver, CO
Contact: William C. Healy, Jr., Marathon Oil Co., P. O. Box 269, Littleton, CO 80160; 303/794-2601

October 15-18
SOFTCON/FALL
Los Angeles, CA
Contact: Northeast Expositions, 822 Boylston Street, Chestnut Hill, MA 02167; 617/739-2000

October 20-24
Computer Graphics Atlanta
Atlanta, GA
Contact: CGA 85 Conference Director, 2033 M Street NW, Suite 333, Washington, DC 20036; 202/775-9556

October 21-23
Intersociety Conference on Artificial Intelligence Applications
Washington, DC
Sponsors: IEEE, ORSA, TIMS, Washington AI group
Contact: B. G. Silverman, Institute for Artificial Intelligence, Library, Room 636, George Washington University, Washington, DC 20052

October 21-23
1985 IEEE Symposium on Foundations of Computer Science
Portland, OR
Contact: Eugene Luks, Computer and Information Science, University of Oregon, Eugene, OR 97403; 503/686-4408

October 23-25
Protext II: Second International Conference on Text Processing Systems
Dublin, Oreland
Contact: Protext II Organizing Committee, c/o Boole Press, Ltd., P. O. Box 5, 51 Sandycove Road, Dun Laoghaire, County Dublin, Ireland: 353-1/808025

October 23-25
IEEE Symposium on Expert Systems in Government
McLean, VA
Contact: Expert Systems Symposium, IEEE, 1730 Massachusetts Avenue NW, Washington, DC 20036-1903; 202/371-0101

October 28-30
DPMA Houston 85
Houston, TX
Contact: Conference Manager, Data Processing Management Association, 505 Busse Highway, Park Ridge, IL 60068-3191; 312/825-8124

October 28-30
Fourth International Conference on Entity-relationship Approach
Chicago, IL
Contact: Jane W. S. Liu, Dept. of Computer Science, 1304 W. Springfield Avenue, University of Illinois, Urbana, IL 61801; 217/333-0135

October 28-30
IEEE International Conference on Computer Languages
Miami Beach, FL
Contact: Joe Urban, University of S. W. Louisiana, Dept. of Computer Science, P. O. Box 44330, Lafayette, LA 70504; 318/231-6304

October 29-31
Sixth Annual Federal Office Automation Conference
Washington, DC
Contact: The Federal Office Automation Conference, P. O. Box N, Wayland, MA 01778; 617/358-5356

NOVEMBER

November 13-15
3rd International Conference on Optical Memories, Video Disks, and Compact Disks of the Year 2000
San Francisco, CA
Contact: Judith P. Hanson, TOC Coordinator, Technology Opportunity Conference, P. O. Box 14817, San Francisco, CA 94114; 415/626-1133

November 18-22
IEEE Tutorial Week—Washington 1985
Arlington, VA
Contact: Director of Tutorials, IEEE, 1730 Massachusetts Avenue NW, Washington, DC 20036-1903; 202/371-0101

November 20-24
COMDEX
Las Vegas, NV
Contact: The Interface Group, Registration Dept., 300 First Avenue, Needham, MA 02194; 617/449-6600

INDEX TO PRODUCTS

PERFECT RECALL

FOR YOUR PC NETWORK

It's easy to back up data from your hard disk to a cartridge tape system.

The trick is getting it back again.

The ADIC Data Library gives you the most error free backup system available. Its unique redundant recording and error detection/correction technique and internal microprocessor reconstruct detected errors, providing an error rate of less than one in 10^{10} bits.

ADIC has been providing our customers with error correction technology for over 3 years.

The ADIC Data Library is the only tape backup system to use preformatted tape cartridges which provides true random access to files. The Data Library fully emulates a hard disk, using all standard MS DOS commands. And ADIC's version of LARCHIVE and LRESTORE is designed for the IBM, Novell, and other Networks.

ADIC's Data Library is a true DOS device, ready to store or retrieve data as you would from your floppy or hard disk drive, without software modification.

The ADIC Data Library is the only system that can guarantee tape interchange from system to system. Head to tape alignment, gain setting, tape speed, and tape tensioning are all controlled by microprocessor. This completely eliminates all mechanical and electronic adjustments.

You back up your data because it's valuable. Just one unrecoverable file can ruin your whole day. With the Data Library, you know you'll get your data back when you need it.

67 to 268 Mbyte (formatted) systems are available to meet your backup needs.

The ADIC Data Library – for perfect reCall 1-800-638-0818.

adic

Advanced Digital Information Corporation

P.O. Box 2996
10201 Willows Road
Redmond, Washington 98073
(206) 881-8004
Toll Free 1-800-638-0818

CIRCLE NO. 248 ON READER SERVICE CARD

AT&T TeleVideo EAGLE COMPAQ COLUMBIA DATA PRODUCTS, INC.
TANDY SPERRY apple ZENITH
IBM LEADING EDGE

We're judged by the compatibility we keep!

The INFAX Stack™

In addition to being totally compatible with over 20 of the leading computers manufactured today, the **INFAX Stack** offers many other advantages.

The **INFAX Stack** features the convenience of removable disk data cartridges. Each data cartridge has 10 megabytes of formatted capacity and over 14 megabytes unformatted. Whether you're using the **INFAX Stack** for primary storage or for backup, you can have infinite power to spare.

The **INFAX Stack** eliminates the need to put a hodgepodge of applications on a single, do-it-all, fixed disk. The **INFAX Stack** also excludes the need to backup floppy disks and the time wasted.

The **INFAX Stack** features Winchester speed coupled with unmatched dependability. The **INFAX Stack** is a rugged sub-system designed for years of reliable operation, thousands of hours of use have proved this true.

In fact, head crashes are a thing of the past. The **INFAX Stack** uses the time tested Bernoulli technology and the aerodynamics associated with it.

The **INFAX Stack** comes complete with everything you need to tailor it to match your PC's exact configuration. In addition to twin disk cartridges (each approximately 8½″×11″), the **INFAX Stack** comes complete with data cable, adaptor card and applicable software.

The **INFAX Stack** can be up and running in less than 15 minutes. The **INFAX Stack** is user-friendly and menu driven. So put your good judgement to use and choose the **INFAX Stack.** Call or write for the name of your nearest INFAX dealer. It's the compatible thing to do!

APPLE, AT&T, COLUMBIA, COMPAQ, EAGLE, IBM, LEADING EDGE, SPERRY, TANDY, TELEVIDEO and ZENITH are all registered trademarks of their respective companies.

INFAX®

INFAX INCORPORATED
5301 Covington Highway
Decatur, Georgia 30035 (404) 981-6778

CIRCLE NO. 151 ON READER SERVICE CARD